FEAP - - A Finite Element Analysis Program

Version 8.5 Isogeometric User Manual

Robert L. Taylor & Sanjay Govindjee
Department of Civil and Environmental Engineering
University of California at Berkeley
Berkeley, California 94720-1710

Revised October 2017

Contents

List of Figures

List of Tables

Chapter 1

INTRODUCTION

Isogeometric analysis, introduced by Hughes *et al.*[1], is gaining popularity as an alternative to traditional finite analysis methods. In an isogeometric analysis the interpolation functions use NURBS[2] or a variation thereof instead of the more traditional Lagrange interpolation or other types of local polynomial approximation. Both rely on use of parametric interpolation to represent both coordinates and the dependent variables. Both also commonly use an isoparametric interpolation to map element shapes between the parent and global coordinate domains. Isogeometric methods permit the use of approximations that can be C^p continuous in the analysis domain. The method is described fully in the book by Cottrell *et al.*.[3] Additional information on isogeometric analysis may be found in References [4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23]

FEAP has been adapted to permit the specification of the analysis region using *tensor product NURBS patches*. The current implementation includes a library of elements capable of solving problems in solid and structural mechanics. In addition thermal problems may also be solved.

The description of the data to solve a thermal, solid or structural mechanics problem utilizing *FEAP* is described in the companion User Manual[24]. Many of the commands necessary to solve a problem using an isogeometric description are the same. However, there are some important differences existing and this manual serves as a supplement to the *FEAP* User manual. All manuals are maintained on-line at the web site

```
projects.ce.berkeley.edu/feap
```

and should be consulted from time-to-time to obtain any description of new features.

1.1 NURBS interpolation on a line

Before describing how the data is prepared for an isogeometric analysis using *FEAP* it is important to understand how parametric interpolation is performed. Here we discuss the basic aspects using a one-dimensional interpolation along a line. The basic forms include use of a parametric description in terms of a parent coordinate, u. The parametric coordinate is used to describe a *knot vector* with m values. In the current *FEAP* implementation only *open knot vectors* are implemented (this could change in future). An open knot vector is described by an increasing sequence of values. For example one may have the knot vector with $m = 8$ values

$$\mathbf{U} = \begin{bmatrix} 0 & 0 & 0 & 0.25 & 0.8 & 1 & 1 & 1 \end{bmatrix}$$

The first an last values of an open knot vector are repeated m times and will describe a polynomial of degree $p = m - 1$. Thus the above knot vector is capable of describing a quadratic polynomial. Knot vectors for which the individual knots are placed at equal intervals along the parent coordinates are called *uniform knot vectors*. Those in which the intervals vary are called *non-uniform knot vectors*. In isogeometric analysis increments along the knot vector describe individual *element* intervals. Thus the above knot vector would describe three element intervals: [0 0.25]; 0.25 0.8; and [0.8 1]. The introduction of each knot lowers the continuity of the polynomial interpolation by *one (1)*. Thus, at the location of the knots 0.25 and 0.8 the continuity is reduced from two to one. Thus, the second derivative of the (as yet undefined polynomial function) will have a slope discontinuity at the knot points. If an additional knot is *inserted* at 0.25 giving the new knot vector

$$\mathbf{U} = \begin{bmatrix} 0 & 0 & 0 & 0.25 & 0.25 & 0.8 & 1 & 1 & 1 \end{bmatrix}$$

no new element interval is created (i.e., there are still only three element intervals) but the continuity is reduced to degree 1 at 0.25 and a slope discontinuity may now exist in the first derivative.

An polynomial function for a knot vector may be created using B-splines[2]. A B-spline may be described starting with

$$B_{i,0}(u) = \begin{cases} 1; & \text{if} u_i \le u < u_{i+1} \\ 0; & \text{otherwise} \end{cases}$$

followed by the recursion

$$B_{i,p}(u) = \frac{u - u_i}{u_{i+p} - u_i} B_{i,p-1}(u) + \frac{u_{i+p+1} - u}{u_{i+p+1} - u_{i+1}} B_{i+1,p-1}(u)$$

Note the interpolation is described in terms of the knot locations of $\mathbf{U}$ and each evaluation of the recursion adds one additional function. In the recursion the ratio $0/0$ can occur and is defined as 0. The basis functions $B_{i,0}$ are piecewise constants. Thus the first recursion will create piecewise continuous linear polynomials.

Example 1:

Consider the knot vector $\mathbf{U} = \begin{bmatrix} 0 & 0 & 1 & 1 \end{bmatrix}$. This has the zeroth order basis functions

$$\begin{aligned} N_{0,0} &= 0 \;\; ; \quad -\infty < u < \infty \\ N_{1,0} &= \begin{cases} 1 \;\; ; & 0 \le u < 1 \\ 0 \;\; ; & \text{otherwise} \end{cases} \\ N_{2,0} &= 0; \quad -\infty < u < \infty \end{aligned}$$

Applying the recursion formula gives

$$\begin{aligned} N_{0,1} &= \frac{u-0}{0-0} N_{0,0} + \frac{1-u}{1-0} N_{1,0} = \begin{cases} 1-u \;\; ; & 0 < u < 1 \\ 0 \quad ; & \text{otherwise} \end{cases} \\ N_{1,1} &= \frac{u-0}{1-0} N_{1,0} + \frac{1-u}{1-1} N_{2,0} = \begin{cases} u \;\; ; & 0 < u < 1 \\ 0 \;\; ; & \text{otherwise} \end{cases} \end{aligned}$$

These are identical to the linear Lagrange polynomials and are only C_0 continuous. The open knot vector of a C_0 function has only two repeated first and last entries. Results using such a form in an isogeometric formulation will produce identical results to linear order Lagrange elements. Thus an isogeometric analysis usually implies use of quadratic an higher order description of open knot vectors.

The interpolation of the coordinates along the line is given by

$$\mathbf{x} = \sum_{i=1}^{n} B_{i,p}\, \tilde{\mathbf{x}}_i \quad \text{where} \quad n = m - p - 1$$

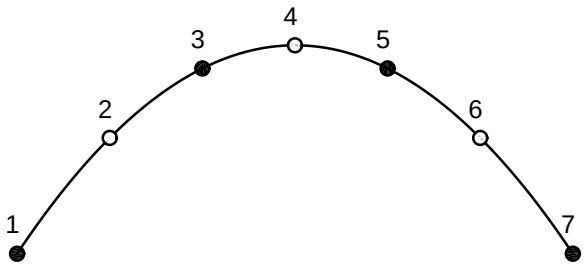
The parameters $\tilde{\mathbf{x}}_i$ are called *control points* and take the place of the *nodes* of a traditional finite element analysis. There are fundamental differences between control points and nodes.

Example 2:

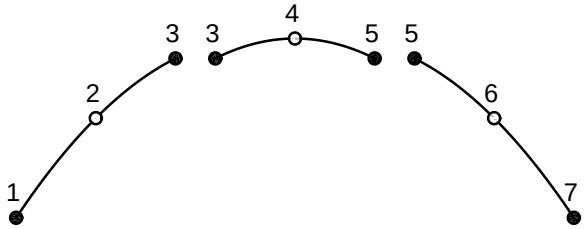
As an example let us consider the description of a parabolic line using quadratic degree interpolation and three equal size elements in the parametric domain. For a standard finite element interpolation we use the three Lagrange shape functions[25]

$$\begin{aligned} N_1(\xi) &= \tfrac{1}{2}(\xi^2 - \xi) \\ N_2(\xi) &= \tfrac{1}{2}(\xi^2 + \xi) \\ N_3(\xi) &= (1 - \xi^2) \end{aligned}$$

The specific parabola is defined on the interval $-12 \le x \le 12$ which has altitude $y = 9$ at $x = 0$ and end values $y = 0$ at $x = \pm 12$. The location of the nodes for a Lagrange



(a) Parabolic curve with 3 quadratic elements



(b) Split of curve into individual elements

Figure 1.1: Lagrange interpolation of a parabola

interpolation are placed at 7 points equally spaced along the x-axis with values

$$\tilde{\mathbf{x}}_1 = \begin{Bmatrix} -12 \\ 0 \end{Bmatrix} \;\; ; \;\; \tilde{\mathbf{x}}_2 = \begin{Bmatrix} -8 \\ 5 \end{Bmatrix} \;\; ; \;\; \tilde{\mathbf{x}}_3 = \begin{Bmatrix} -4 \\ 8 \end{Bmatrix} \;\; ; \;\; \tilde{\mathbf{x}}_4 = \begin{Bmatrix} 0 \\ 9 \end{Bmatrix}$$
$$\tilde{\mathbf{x}}_5 = \begin{Bmatrix} 4 \\ 8 \end{Bmatrix} \;\; ; \;\; \tilde{\mathbf{x}}_6 = \begin{Bmatrix} 8 \\ 5 \end{Bmatrix} \;\; ; \;\; \tilde{\mathbf{x}}_7 = \begin{Bmatrix} 12 \\ 0 \end{Bmatrix}$$

A plot of the line using the Lagrange interpolation is shown in Fig. 1.1. The C_0 nodes are shown as black dots and the internal nodes for N_3 by white dots. In the (b) part of the figure we show the individual elements and their associated nodes.

Next we consider the same parabola where the interpolation is performed using quadratic B-spline interpolation. To create three elements we use the knot vector

$$\mathbf{U} = \begin{bmatrix} 0 & 0 & 0 & \frac{1}{3} & \frac{2}{3} & 1 & 1 & 1 \end{bmatrix}$$

Applying the recursion formula generates creates only 5 unique functions as shown in Fig. 1.2. The location of the control points to construct the desired parabola are located at

$$\tilde{\mathbf{x}}_1 = \begin{Bmatrix} -12 \\ 0 \end{Bmatrix} \;\; ; \;\; \tilde{\mathbf{x}}_2 = \begin{Bmatrix} -8 \\ 6 \end{Bmatrix} \;\; ; \;\; \tilde{\mathbf{x}}_3 = \begin{Bmatrix} -0 \\ 10 \end{Bmatrix} \;\; ; \;\; \tilde{\mathbf{x}}_4 = \begin{Bmatrix} 8 \\ 6 \end{Bmatrix} \;\; ; \;\; \tilde{\mathbf{x}}_5 = \begin{Bmatrix} 12 \\ 0 \end{Bmatrix}$$

These produce the parabola shown in Fig. 1.3.

There are distinct differences between B-spline interpolation and Lagrange interpolation. Except for the end points of an open knot the control points do not lie on the

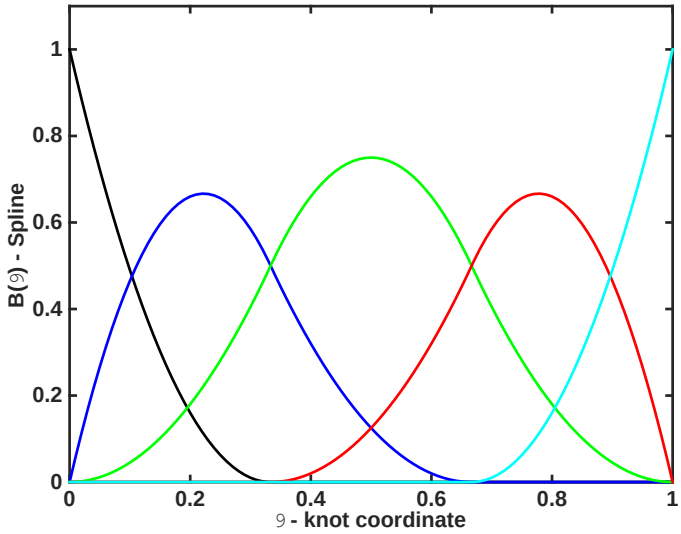
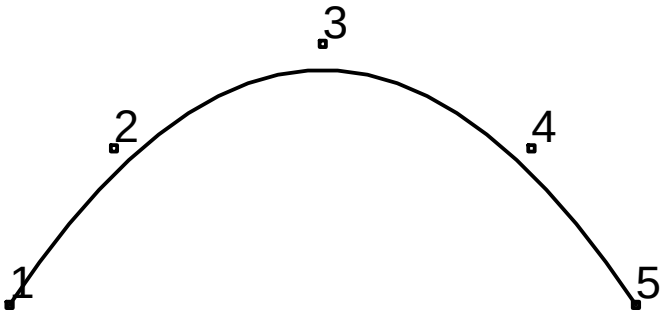
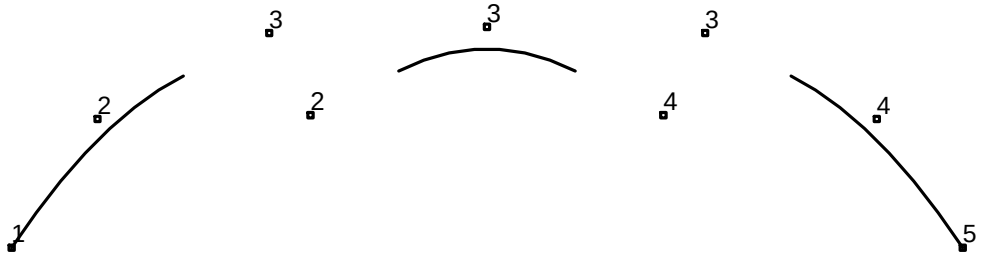


Figure 1.2: Quadratic B-spline for quadratic knot with three elements. Generates 5-functions.

curve of the parabola. Moreover, the description of each element involves control points that overlap between elements – this is what allows for the increased continuity. Last,



(a) Quadratic B-spline curve with 5 control point positions.



(b) Quadratic B-spline elements.

Figure 1.3: B-spline curve for parabola.

we observe that the B-spline functions are positive everywhere. The sum of all the B-splines of a order p always equals one, thus B-splines are a *partition of unity*. There are other aspects related to the use of spline interpolation, these are discussed in the references cited above.

1.1.1 NURBS functions and Bézier extraction

Non-uniform rational B-splines (or simply NURBS) utilize an additional parameter to describe the control point – these are positive weights w_i. The NURBS functions are defined as

$$N_{i,p}(u) = \frac{B_{i,p}(u)w_i}{\sum_{j=1}^{n} B_{j,p}(u)w_j}$$

Use of the weights allows for shapes of different types, including circular arcs. If all the weights are unity, the denominator sums to one and B-splines are recovered.

The construction of the B-spline functions using the recursion formula is awkward and can become time consuming for high-degree functions. An alternative is to use Bézier extraction to relate the B-spline functions in each non-zero knot interval to Bernstein polynomials. This is described in detail by Borden *et al.*[26]. For each knot interval a $p+1 \times p+1$ matrix $\mathbf{C}^e$ may be described such that

$$B_{i,p}(\xi) = C^e_{ij} b_{j,p}(\xi)$$

where the $b_{j,p}$ are Bernstein polynomials. For quadratic polynomials on the element interval $-1 \leq \xi \leq 1$ the Bernstein polynomials are

$$\begin{aligned} b_{1,2} &= \tfrac{1}{4}(1-\xi)^2 \\ b_{2,2} &= \tfrac{1}{4}(1+\xi)^2 \\ b_{3,2} &= \tfrac{1}{2}(1-\xi^2) \end{aligned}$$

Use of Bézier extraction greatly simplifies the construction of shape functions once the *extraction matrices* $\mathbf{C}^e$ are known. The extraction matrices are defined by a simple *knot insertion* algorithm as described in Piegl & Tiller[2].

Higher dimensional interpolation may be defined by taking products of the one-dimensional form in each desired coordinate direction. These are called *tensor product* forms and currently form the basis for nearly all the developments currently available in *FEAP*.

With this brief set of preliminaries, we now describe how the data is prepared for a NURBS based solution using *FEAP*.

Chapter 2

NURBS mesh description

We describe how to define a tensor product NURBS patch for a *FEAP* analysis. Tensor product patches may be described for one, two or three dimensional applications.

2.1 Control information: Start of problem

The start of an analysis begins with the standard *FEAP* control information. However, for some element forms special care is needed in setting the number of degrees of freedom (NDF). The the control data consists of two lines:

```
FEAP * * <any description of the problem>
  NUMCP NUMEL NUMMAT NDM NDF NEN
```

where

```
NUMCP  = Number of control points
NUMEL  = Number of elements
NUMMAT = Number of material sets
NDM    = Mesh spatial dimension
NDF    = Maximum number of degrees of freedom/control point
NEN    = Maximum number of control points on an element
```

Using the data preparation approach described below for an isoparametric analysis the number of control points (NUMCP), elements (NUMEL) and size of an element (NEN) are generally not known at the start of an analysis. This is due to the specified order of elevation of the knot vector and/or the number of inserted knots; each of which are

operations that may be performed using *FEAP* command instructions described in later sections of this report. Accordingly, these values should be set to zero, *FEAP* will assign values as the mesh for the problem is constructed.

For solution of problems using the displacement form of *solid* elements the value of `NDM = NDF` (or more). For *thermal* analyses the values of `NDM` is the spatial dimension of the problem and `NDF = 1` (or more). These are identical to the usual finite element analysis values as described in the *FEAP* User manual[24].

For `MIXED` solid elements (those based on a $\mathbf{u}$-p-θ formulation as described in references [25] and [23]) the value of `NDF` must be set to `NDM+1`. Thus for a three dimensional analysis using the mixed solid elements the control data is input as

```
FEAP * * <title information for mixed analysis>
  0 0 0 3 4 0
```

For analyses using the Kirchhoff-Love thin element the parameters are set as $NDM = 3$ and $NDF = 3$ or more. Thus for the shell the control records are set as

```
FEAP * * <title information for shell analysis>
  0 0 0 3 3 0
```

Note it is not necessary to provide the number of nodes, elements, material sets or nodes/element. These will be determined based on the subsequent input data provided.

Alternatively, the above statements now may be given for the mixed solid as

```
FEAP * * <title information for mixed solid analysis>
  ndm = 3
  ndf = 4
```

and for the shell as

```
FEAP * * <title information for shell analysis>
  ndm = 3
  ndf = 3
  nad = 2
```

If the solid is encased in a shell the parameters are given as

```
FEAP * * <title information for solid & shell analysis>
  ndm = 3
  ndf = 4
  nad = 2
```

2.2 MATERIAL set description

The specification of material properties generally follows the descriptions described in the *FEAP* User Manual[24]. The types of elements available are:

```
SOLId           - Solid DISPlacement or MIXEd types
THERmal         - Fourier heat conduction
NURB_THIN_SHELL - Kirchhoff-Love thin shell
NURB_SLOPE      - Slope enforcement for K-L shell
PRESsure        - Pressure or traction loading on surface
USER enum
```

Note that the required specification for the element type must include all the letters given in upper case above.

A NURBS solution may be used for both small and large displacement solid elements of type `DISPlacement` or `MIXEd` only. The thin Kirchhoff-Love shell formulation is specified by `NURB_THIN_SHELL` for both the large and small displacement formulation.

2.2.1 Activation of NURBS interpolations

Activation of the NURBS option is given during the specification of the `MATErial` property data by including the statement

```
NURBs interpolation q1 q2 q3
```

as part of the data specification. The parameters `q1`, `q2`, `q3` denote quadrature order in each direction of the tensor product patch. For two dimensional problems it is not necessary to specify `q3`.

2.2.2 SOLId element material sets

To solve a problem using `SOLId` type elements the material data set is given as:

```
MATErial ma
  SOLId
    <ELAStic, PLAStic, VISCoelastic, etc. material model>
    NURBs <option> q1 q2 q3
     ! Blank end record
```

Currently, the `option` parameter is not used. In future it will be used to specify the type of interpolation form (*T-spline*, etc.). The values of `q1`, `q2` and `q3` are the number of quadrature points in the 1,2 and 3 directions (currently, between 1 and 5).

Specific forms of data for constitutive models, body forces, etc. are described in the *FEAP* User Manual.[24]

2.2.3 THERmal element material sets

To solve a problem using the `THERmal` type elements the material data set is given as:

```
MATErial ma
  THERmal
    FOURier <isotrop, orthotropic ...>
    <additional data such as DENSity ...>
    NURBs <option> q1 q2 q3
     ! Blank end record
```

The values of `q1`, `q2` and `q3` are the number of quadrature points in the 1,2 and 3 directions (currently, between 1 and 5).

Description of the material models, etc. is again in the *FEAP* User Manual.

2.2.4 NURB_THIN_SHELL element material sets

To solve a problem using Kirchhoff-Love thin shell type elements the material data set is given as:

```
MATErial ma
  NURB_THIN_SHELL
    <ELAStic, PLAStic, VISCoelastic, etc. material model>
    THICkness SHELL h
    NURBs <option> q1 q2 q3
    <FINIte, SMALl>
     ! Blank end record
```

Here the values of `q1` and `q2` are the number of quadrature points in the 1 and 2 directions of the surface patch describing the element (currently, between 1 and 5). The value of `q3` is the number of quadrature points in the shell thickness direction and must be 2 or more. The commands `FINIte` and `SMALl` are used to denote the large

displacement and small displacement forms, respectively. However, if the material model applies only to finite deformation, for example

```
ELAStic NEOHook E_mod nu
```

then the FINIte is automatically selected.

Descriptions for the material models, body loading, etc. are again in the *FEAP* User Manual (e.g., see Chapter 7 for material model types available).

2.2.5 NURB_SLOPE element material sets

The constraint of slope between NURBS patches is enforced by the user element ELMT27 and is accessed using the material set commands

```
MATErial ma
  NURB_SLOPE
    PENAlty,,pen_value
    QUADrature <NODE, NODAl, GAUSs>
     ! Blank end record
```

The number of quadrature points for all forms is 2. In addition, the number of degree-of-freedoms at a node must be increase to 4 in order to provide storage for the constraint forces. The constraint maintains the initial angle defined by the two patches during the entire solution based on control points only that define a 6-node constraint element. Note, the slope enforcement involves a non-linear relation, thus, a non-linear solution method is required using, for example

```
LOOP,,20 ! or some number
  TANG,<LINE>,1
NEXT
```

2.2.6 PRESsure: Dead and Follower loads

The *pressure load element* is specified by material set records:

```
MATErial ma
  PRESsure
    LOAD p prop-ld
```

ELMT	Description
1	NURBS Euler-Bernoulli beam
2	NURBS 1-d rod.
3	NURBS 1-d displacement boundary condition
5	NURBS & T-spline thin C^1 plate
6	NURBS & T-spline thin membrane
20	Follower couple to load thin shell element
24	NURB_THIN_SHELL - Kirchhoff-Love shell
37	NURB_SLOPE - Slope compatibility enforcement

Table 2.1: User elements for NURBS/T-spline solutions.

```
 NURBs quadr q1 q2
<PLOT,NOPLot> !  PLOT/NOPLot surface: Default NOPLot
<PLANe,AXISym> ! 2-d types: Default PLANe
<DEAD,FOLLower>! Default DEAD
   ...
```

Loading is specified by options `LOAD` and, for follower loads by `FINI`te or `FOLL`ower. Loading intensity may be associated with the proportional loading number `prop-ld`. The `NURBs` option specifies the quadrature order to use in the two surface directions of a 3-d problem. For 2-d problems the second value is not used since the surface has only one-dimension.

2.2.7 USER element material sets

User elements are also provided but vary with specific releases. The basic input form is

```
MATErial ma
  USER   e_num
    <user element data>
     ! Blank end record
```

The `e_num` parameter is the number of the specific user element used. That is if `ELMT04.f` is used then `e_num = 4`.

A number of user elements have been added as described in Table 2.1.

2.3 NURBS patch input

Tensor product NURBS patches are defined by specifying the *control point* data using the NURB command; the *knot vector* in each direction using the KNOT command; and the NPATch command which includes a list of control points to define the patch. Descriptions for specifying a patch for a line, surface or solid is described below.

Alternatively, the patch may be defined using the NBLOck command and the NSIDe command as described in Appendix B. Descriptions for specifying each of these data types is provided below.

A NURBS patch may be specified using the minimum number of control points, order of knot vectors and sides necessary to describe the exact geometry. This description may be subsequently refined by raising the order of the knot vectors (*knot elevation*) and specifying additional knot points (*knot insertion*) necessary to produce an accurate answer (this is commonly called a *k-refinement*).

2.3.1 NURBS control point specification

The control points for a patch are described by the NURBs data. Input of the control points uses the same command structure as for input of COORdinate data with added need to specify the *weight* for the control point. Data sets are input as:

```
NURBs
  n1  ng1  (x(i,n1),i = 1,ndim) w(n1)
  n2  ng2  (x(i,n2),i = 1,ndim) w(n2)
   ....
   ! terminate with blank record
```

where ndim is the mesh spatial dimension and ng1, ng2 are increments to the control point numbers. For example, the two pairs shown above will generate the sequence of control points

```
n1,  n1+ng1, n1+2*ng1, ..... , n2
```

with values for coordinates and weights linearly interpolated between the two specified values.

2.3.2 KNOT vector specification

Currently four types of knot vectors may be used to construct NURBS patches: CLAMped (open) knot vectors; UNCLamped knot vectors; LCLAmped knot vector which is clamped at the start values and unclamped at the end value; and RCLAmped which is unclmaped at the start value and clamped at the end value. Clamped (open) knot vectors are interpolatory at an end control point whereas unclamped knot vectors are not interpolatory unless the knot vector is repeated to give a C_0 point. However, unclamped knot vectors may be used to create closed surfaces or to maintain greater than C_0 continuity between patches.

CLAMped (OPEN) knot vectors

Open knot vectors are specified by the records:[1]

```
KNOTs
  CLAMp n1 lknot1 (vk1(i),i=1,lknot1)
  CLAMp n2 lknot2 (vk2(i),i=1,lknot2)
   ....
    ! terminate with blank record
```

where `n1` is the knot number, `lknot1` is the length of the knot vector and `vk1(i)` is the list of open knot values. Recall that only 16 items can appear on any record, thus, if the knot vector has more than 13 data entries the next 16 appear on the following line, etc. until all the values are provided. Open knot vectors must begin and end with repeated values of one more than the order of the knot. An example for two quadratic knot vectors is

```
CLAMp 1 6 0.0 0.0 0.0 3.0 3.0 3.0
CLAMp 2 7 0.0 0.0 0.0 0.5 1.0 1.0 1.0
```

Generally, one may start the knot at 0.0 and go to any end value desired. However, the knot values *must* appear in ascending order.

The specification of real values for *FEAP* do not need to include the decimal point. Thus, the above knot vectors may also be given in the simpler form

```
CLAMp 1 6  0 0 0 3 3 3
CLAMp 2 7  0 0 0 0.5 1 1 1
```

UNCLamped (periodic) knot vectors

An unclamped knot vector is specified in *FEAP* using the input form

[1] A clamped knot may also be specified by the type OPEN.

```
KNOTs
  UNCLamp n1 lknot1 order1 (vk1(i),i=1,lknot1)
  UNCLamp n2 lknot2 order2 (vk2(i),i=1,lknot2)
   ....
    ! terminate with blank record
```

Similarly the mixed types may be specified as

```
KNOTs
  LCLAmp n1 lknot1 order1 (vk1(i),i=1,lknot1)
  RCLAmp n2 lknot2 order2 (vk2(i),i=1,lknot1)
```

For an `LCLAmp` type the initial values of the `vk1(i)` array must be repeated for `i=1, order1+1` times; and vice versa for the `RCLAmp` type (the last `order2` of the `vk2(i)` array must have the same value).

It is also possible to mix types within the same `KNOTs` group (i.e., some may be `open` while others are `uncl`). Note that an extra field is required to set the order since knot vector values are not repeated at the beginning and end of the `vk1(i)` sequence. An unclamped knot vector may be used to create a closed surface with C_p continuity (where p is one less than the order) by setting the coordinates of the start and end `order` control points to the same value. For this case the beginning and ending overlapped knot spacings must be the same.

In general the `UNCLamped`, `LCLAmp` and `RCLAmp` types in *FEAP* can not create conic sections.

To close a surface using unclamped knots it is necessary to start with uniform knot spacing and *overlap* the last `order1` control points – that is they must have the same coordinate values. To merge (join) the control points of the mesh to have the same node number, the standard

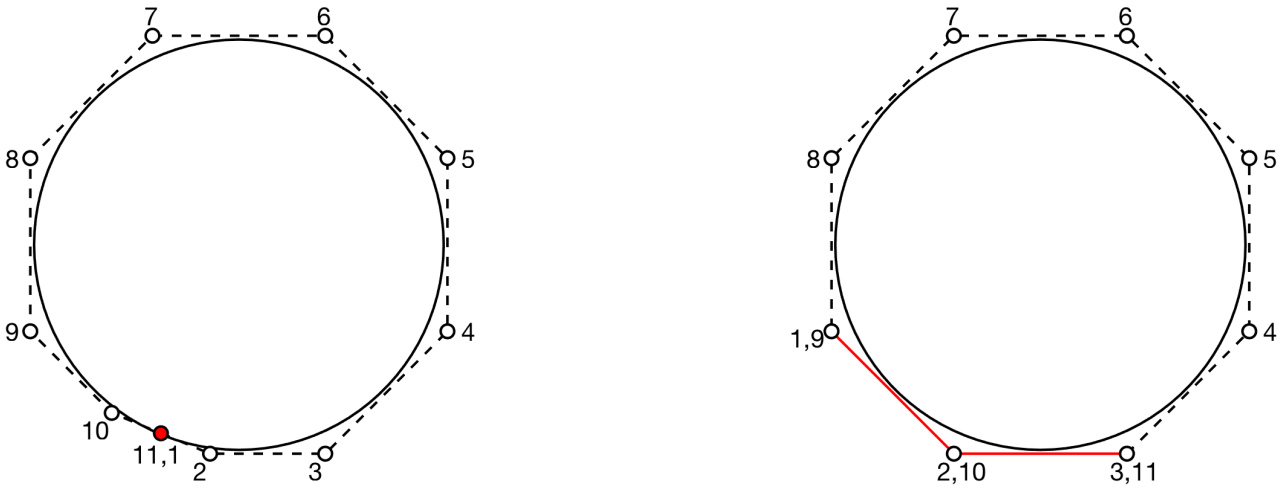
```
TIE
```

command is used (see User Manual for details[24]). *Always use a graphical check for the mesh input to ensure that surfaces actually are closed.*

Example: Closed ring To illustrate the use of clamped and unclamped knows in creating a closed ring we consider the shapes shown inf Fig. 2.1. The left figure shows the control polygon and ring (which is a not a perfect circle since all the control weights are unity). The red marked control point is a location that can only be C_0 (after a `TIE`). The right figure shows the same ring using an *unclamped knot vector*. The red marked portion denotes control points that are over-lapped to preserve the continuity (again after a `TIE`). Figure 2.2 shows the spline functions for each of clamped and unclamped knots. There are eleven (11) functions for each form, however,

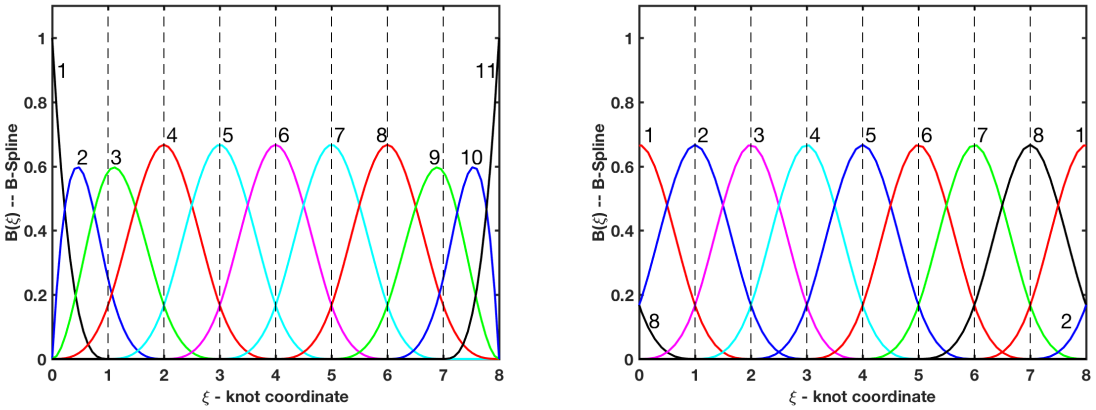
due to the overlap of the control points three (3) of the functions are in fact continuations as the number indicate. The data for the clamped knot and control points is given by:

```
NURBs
  1 0 -4.5949E+01 -1.1093E+02 1.0000E+00
  2 0  1.6973E+01 -1.2293E+02 1.0000E+00
  3 0  5.0920E+01 -1.2293E+02 1.0000E+00
  4 0  1.2293E+02 -5.0920E+01 1.0000E+00
  5 0  1.2293E+02  5.0920E+01 1.0000E+00
  6 0  5.0920E+01  1.2293E+02 1.0000E+00
  7 0 -5.0920E+01  1.2293E+02 1.0000E+00
  8 0 -1.2293E+02  5.0920E+01 1.0000E+00
  9 0 -1.2293E+02 -5.0920E+01 1.0000E+00
 10 0 -7.4925E+01 -9.8929E+01 1.0000E+00
 11 0 -4.5949E+01 -1.1093E+02 1.0000E+00

KNOTs
  CLAMp 1 15 0 0 0 0 1 2 3 4 5 6 7 8 8
    8 8  ! limit of 16 items/record

NPATch
  LINE  1 11 1
   1 2 3 4 5 6 7 8 9 10 11
```



(a) Clamped knot (b) Unclamped knot

Figure 2.1: Close ring using clamped and unclamped knot vector. The *red* control point of (a) is C_0 and the *red* overlap of (b) maintains the C_2 continuity of the cubic curve.



(a) Clamped B-splines (b) Unclamped B-splines

Figure 2.2: Close ring B-splines for clamped and unclamped knot vector.

Similarly, for the unclamped case the data is given by

```
NURBs
  1 0 -1.2293E+02 -5.0921E+01  1.0000E+00
  2 0 -5.0921E+01 -1.2293E+02  1.0000E+00
  3 0  5.0921E+01 -1.2293E+02  1.0000E+00
  4 0  1.2293E+02 -5.0921E+01  1.0000E+00
  5 0  1.2293E+02  5.0921E+01  1.0000E+00
  6 0  5.0921E+01  1.2293E+02  1.0000E+00
  7 0 -5.0921E+01  1.2293E+02  1.0000E+00
  8 0 -1.2293E+02  5.0921E+01  1.0000E+00
  9 0 -1.2293E+02 -5.0921E+01  1.0000E+00
 10 0 -5.0921E+01 -1.2293E+02  1.0000E+00
 11 0  5.0921E+01 -1.2293E+02  1.0000E+00

KNOTs
  UNCL 1 14 3 0 1 2 3 4 5 6 7 8 9 10 11 12
  13 14  ! limit of 16 items/record

NPATch
  LINE  1 11 1
```

2.3.3 NPATch specification

One dimensional patch

For a one dimensional block the command is given as

```
NPATch
  LINE ma np kn
    cp(1) cp(2) .... cp(np)
```

where `ma` is the patch material set number, `np` the number of control points defining the line and `kn` the knot number of the line. The list `cp` defines the control point numbers.

Example: 1-d NURBS Patch

As an example consider again the parabola shown in Fig. 1.1(a). The input data using `NPATch` becomes:

```
NURBs
  1  0 -12.0   0.0  1.0
  2  0   0.0  18.0  1.0
  3  0  12.0   0.0  1.0

KNOTs
  open 1 6 0.0 0.0 0.0 1.0 1.0 1.0

NPATch
  SURFace 1 3 1
    1  2  3
```

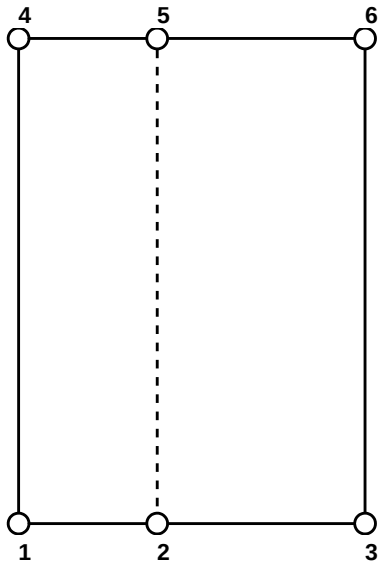


Figure 2.3: Two dimensional NURBS surface patch

Surface patch

Surface patches may be used to define either a two dimensional solids problem or a three dimensional shell surface. The surface patch of NURBS is specified using the command statements

```
NPATch
  SURFace ma np1 np2 kn1 kn2
    cp(1,1) cp(2,1) .... cp(np1,1)
    cp(1,2) cp(2,2) .... cp(np1,2)
       ....
    cp(1,np2) ...........cp(np1,np2)
```

In the above `ma` is the material number for the patch; `np1, np2`the number of control points along the two sides of the mesh; `kn1, kn2` the knot vectors in the two directions of the patch; and `cp(i,j)` are the NURBS control point numbers in which 'i' is in the 1-direction and 'j' is in the 2-direction.

Example: 2-d NURBS Patch

As an example consider again the patch shown in Fig. 2.3. The input form using the `NPATch` option becomes:

```
NURBs
  1  0   0.0   0.0  1.0
  2  0  40.0   0.0  1.0
  3  0 100.0   0.0  1.0
  4  0   0.0 140.0  1.0
  5  0  40.0 140.0  1.0
  6  0 100.0 140.0  1.0

KNOTs
  open 1 4 0.0 0.0 1.0 1.0
  open 2 5 0.0 0.0 0.5 1.0 1.0

NPATch
  SURFace 1 2 3 1 2
    1  4
    2  5
    3  6
```

Solid patch

A three dimensional solid patch may be defined using the commands

```
NPATch
  SOLId ma np1 np2 np3 kn1 kn2 kn3
    cp(1,1) cp(2,1) .... cp(np3,1)
    cp(1,2) cp(2,2) .... cp(np3,2)
       ....
    cp(1,np1) ...........cp(np3,np1)
    cp(1,np1+1) .........cp(np3,np1+1)
       ....
    cp(1,2*np1) .........cp(np3,2*np1)
    cp(1,2*np1+1) .......cp(np3,2*np1+1)
       ....
    cp(1,3*np1) .........cp(np3,3*np1)
    cp(1,3*np1+1) .......cp(np3,3*np1+1)
       ....
       ....
    cp(1,np1*np2) .......cp(np3,np1*np2)
```

Where `ma` is the material number, `np1, np2, np3`, are the number of control points in the three directions of the block, and `kn1, kn2, kn3` are the knot vector numbers in the three directions.

For a simple rectangular block the input data is given by

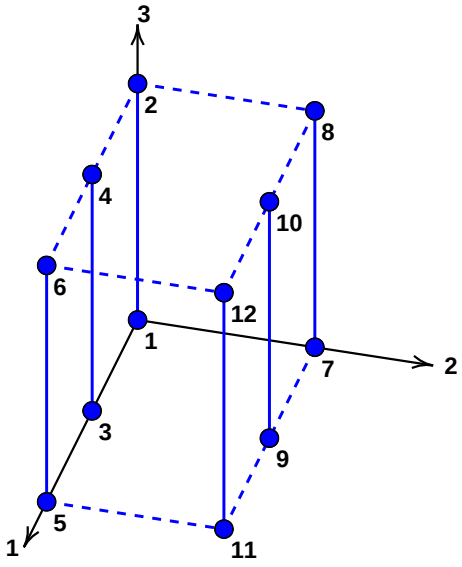


Figure 2.4: Three dimensional NURBS solid patch

```
NURBs
  1  0   0.0   0.0   0.0 1.0
  2  0   0.0   0.0  10.0 1.0
  3  0   5.0   0.0   0.0 1.0
  4  0   5.0   0.0  10.0 1.0
  5  0  10.0   0.0   0.0 1.0
  6  0  10.0   0.0  10.0 1.0
  7  0   0.0   6.0   0.0 1.0
  8  0   0.0   6.0  10.0 1.0
  9  0   5.0   6.0   0.0 1.0
 10  0   5.0   6.0  10.0 1.0
 11  0  10.0   6.0   0.0 1.0
 12  0  10.0   6.0  10.0 1.0

KNOTs
  open 1 6 0.0 0.0 0.0 1.0 1.0 1.0
  open 2 4 0.0 0.0 1.0 1.0
  open 3 4 0.0 0.0 1.0 1.0

NPATch
  SOLId 1  3 2 2  1 2 3
    1  2
    3  4
    5  6
    7  8
    9 10
   11 12
```

2.3.4 Example: Two dimensional curved beam

The complete data for a curved beam loaded by an end shear is given in Table 2.2 and shown in Fig. 2.5.

2.4 Traction surface loading

The application of surface loading by specified traction involves computation of the term

$$\Pi_{ext} = \int_{\Gamma_t} \delta\mathbf{u}\,\bar{\mathbf{T}}\,\mathrm{d}\Gamma$$

for cases where the traction $\bar{\mathbf{T}}$ is specified on the reference configuration. For cases in which the traction is specified on the deformed configuration the loading is obtained from

$$\Pi_{ext} = \int_{\gamma_t} \delta \mathbf{u}\, \bar{\mathbf{t}}\, \mathrm{d}\gamma$$

and then often also requires computation of a tangent term.

At present *FEAP* includes only the first option for some special cases.

2.4.1 NSURface loading

The NSURface loading option is restricted to normal loading applied to *straight* edges of two dimensional NPATch region. The can be specified by a linear or quadratic Lagrange interpolation between specified end points. For linear variation the data is given as

```
NSURface
  SIDE LINEar nside
    1  x1  y1 p1
    2  x2  y2 p2
```

and for quadratic variation by

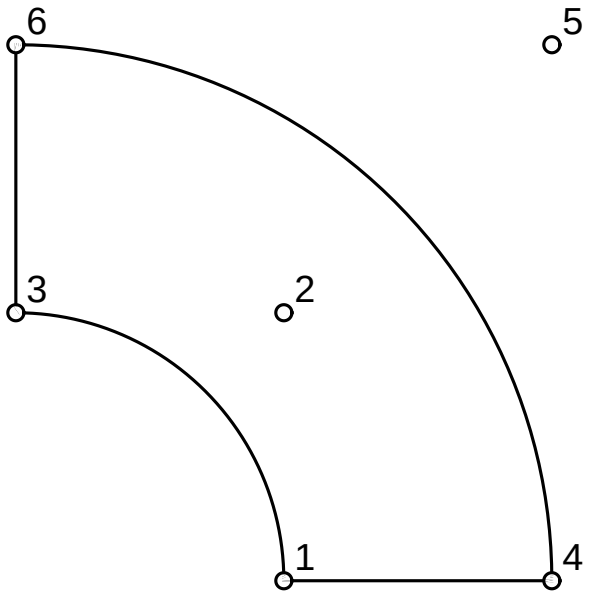


Figure 2.5: Curved beam mesh description.

```
FEAP * * Curved beam NURBs solution
  0 0 0 2 2 0

MATE
  solid
    elastic isotropic 1.e5 0.25
    nurb    interp      3    3

EBOUndary
  1 0 1 0
  2 0 1 0

EDISplacement
  2 0 0.1 0.0

CBOUndary
  node 0 5 1 1

KNOTs
  open 1 4 0.00 0.00 1.00 1.00
  open 2 6 0.00 0.00 0.00 1.00 1.00 1.00

NURBs
  1 0  5.0  0.0  1.00
  2 0  5.0  5.0  1/sqrt(2)
  3 0  0.0  5.0  1.00
  4 0 10.0  0.0  1.00
  5 0 10.0 10.0  1/sqrt(2)
  6 0  0.0 10.0  1.00

NPATch
  SURFace 1  3 2  2 1
    1 2 3
    4 5 6

END
```

Table 2.2: Input data for 2-d curved beam.

```
NSURface
  SIDE QUADratic nside
    1  x1  y1 p1
    2  x2  y2 p2
    3  x3  y3 p3
```

where `x3, y3` is an intermediate point between `x1, y1` and `x2, y2`. The parameter `side` refers to a specific `NSIDe` number.

2.4.2 NLOAd patch loading

The NLOAd option permits a constant traction loading to be specified on any face of a two or three dimensional NPATch. Table 2.3 indicates how faces are numbered on patches. For two dimensional patches the side numbers also are denoted as shown in Figure 2.6 where the 1 and 2 directions are associated with the knot directions.

This option produces *dead* loading only (*i.e.*, loads associated with the reference configuration). The advantage over use of PRESsure material sets (see Sect. 2.2.6) results from only one computation to compute effective control point forces. Whereas, loads from the pressure set are computed for each iteration in the solution by integration over the affected surface. Pressure set loading, however, can be computed on the current configuration as follower type loads.

Face	2 Dimensions	3 Dimensions
1	+1 knot	+1 knot
2	+2 knot	+2 knot
3	-1 knot	+3 knot
4	-2 knot	-1 knot
5	–	-2 knot
6	–	-3 knot

Table 2.3: Face numbers for NPATch patches.

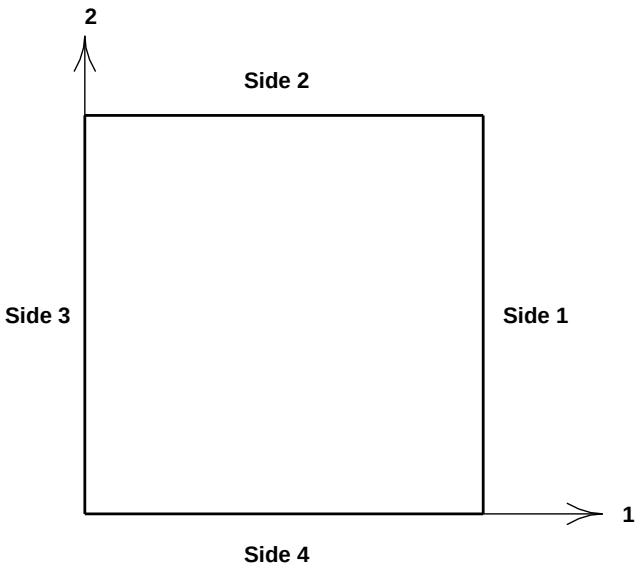


Figure 2.6: Side designation for a two dimensional NURBS patch.

Three options for loading direction are available: `NORMal`, `TRACtion` and `USER`. The `USER` option requires the addition of a user prepared subprogram, although a sample is available for uniform tension in the x_1 coordinate direction of an infinite domain containing a circular hole of radius R. For the `NORMal` option the data is specified as

```
NLOAd
  NORMal patch face pressure prop-num
```

where `patch` is the number of the `NPATch`; `face` is the face number on the patch; `pressure` is the normal traction acting on the face and `prop-num` is a proportional load number for time dependent loading.

For the `TRACtion` option the data is specified as

```
NLOAd
  TRACtion patch face traction direction prop-num
```

where `traction` is the intensity of the specified traction; `direction` the global direction of application and `prop-num` is a proportional load number for time dependent loading.

For the `USER` option the data is specified as

```
NLOAd
  USER patch face value-1 value-2 prop-num
```

where `value-1` and `value-2` are two user available parameters and `prop-num` is a proportional load number for time dependent loading.

2.5 NURBS mesh refinement

Generally, the initial input data defining the geometry of the problem is not sufficient for an accurate analysis. It may be necessary to *elevate* the order of the NURBS approximation for the patches or to *insert* additional knot values in the knot vectors. To perform these steps by hand is a tedious and error prone process. The current implementation in *FEAP* permits both steps to be performed using solution 'Command Language' statements. These are generally given in a batch solution.

2.5.1 Degree ELEVation

To raise the polynomial order of one B-spline defining a NURBS the solution command set

```
BATCh
  ELEVate PATCh blk dir incr
END
```

may be used. The parameters are: `blk` is the patch number; `dir` is the direction in the patch to elevate; and `incr` is the order increment to increase. Execution of these commands creates a file named: `NURBS_mesh` that contains the new set of control points, side lists, knot vectors and patches. At this stage only one direction in one patch has been elevated and it is necessary to repeat the process for other patches and directions. The process of repeating the process can be performed using the *FEAP* input `LOOP-NEXT` commands.

Multiple elevations

In addition to preparing the input file for the original problem description, the process of performing several elevations can be accommodated easily by preparing an additional mesh input file with the structure

```
FEAP <optional title information>
  0 0 0 ndm ndf nel

MATE <all material properties included in original mesh

INCLude NURBS_mesh

END
STOP
```

Then prepare a third input file that has the form:

```
INCLude I<original mesh>
BATCH
  ELEVate PATCh pat1 dir1 incr1
END

INCLude I<NURBS_mesh>
BATCh
  ELEVate PATCh pat2 dir2 incr2
END

etc. for other patches/directions
```

where I<NURBS_mesh> is the name of the second input file containing the INCLude NURBS_mesh statement. The analysis is initiated by specifying the third file as the solution input file. After the processing of the original mesh the subsequent processes may all use the file containing the NURBS_mesh include. Using parameters and looping structures such as

```
PARAmeter
  d = 1

LOOP,2
  PARAmeter
    d = d + 1

  INCLude I<NURBS_mesh>  ! File with INClude NURBS_mesh
  BATCh
    ELEVate PATCh 1 d 2
  END
NEXT
```

would elevate the second and third directions of three-dimensional patch 1 by two orders.

2.5.2 KNOT insertion

To insert knots the command set

```
BATCh
  INSErt PATCh pat dir value rr
END
```

may be use. The parameters are: pat is the patch number; dir the knot direction in the patch; value the location in the knot vector to perform an insertion; and rr the number of times to repeat the insertion.

Each use of an INSErt set again results in a new NURBS_mesh. Multiple insertions can again be performed using the above loop structure. For meshes that perform several knot insertions some time may elapse before the final NURBS_mesh file is created.

After several elevations and insertions, the final NURBS_mesh file contains an isogeometric problem description suitable for analysis. The analysis can be performed using an input file containing the INCLude NURBS_mesh along with boundary conditions and loading specification. This is now a standard *FEAP* solution process and any of the solution options described in the User Manual may be used. Recall that the activation

of a NURBS analysis in an element is controlled by a statement in the MATErial set data:

```
MATE ma
   ....
  NURBs,,q1,q2,q3
   ....
```

2.6 Multiple NURBS blocks

Problems may be solved using multiple NURBS blocks. However, the edge or boundary of contiguous blocks must have the same control point topology – the topology in other directions may be different.

If multiple blocks are used, prior to any solution commands it is necessary to merge the blocks into a single problem using the TIE command. This command appears after the END of mesh command, thus, the general form is:

```
FEAP * * <title information
  0 0 ....
 <mesh data>
END ! End of mesh
TIE
<solution commands>
```

2.6.1 Slope compatibility enforcement for thin shells: NTIE

For the thin shell element use of the TIE command results in a moment-less hinge between the patches. In order to restore slope compatibility between patches it is necessary to add the command NTIE after the TIE command. It is possible enforce slope compatibility between specific patches or to enforce it between all shell patches. To enforce a compatibility between individual patches the command is given as

```
TIE
NTIE
  PATCh p1 p2 ma
```

where p1 and p2 are the two patches and ma is the material set number for the NURB_SLOPE material set (see Section 2.2).

To enforce compatibility between all patches the command is given as

```
TIE
NTIE
  ALL ma
```

where ma is the material set number for the NURB_SLOPE material set (see Section 2.2).

2.7 Surface extraction

In many problems it is necessary to define segments of surfaces from the NURBS patches. These may be found using the solution command N_EXtract. This should be performed in an *interactive mode of solutions*. To initiate the extraction it is necessary to first display a plot of the problem in graphics mode. For two dimensional problems one should first give the command

```
PLOT MESH
```

this is then followed with the command

```
N_EXtract
```

The program will then display each of the boundary segments for each NURBS patch and the user may select to output a file or reject it.

In a three dimensional problem the graphics commands

```
PLOT PERSpecitive
PLOT HIDE
PLOT MESH
```

should be issued prior to giving the

```
N_EXtract
```

command.

After completing the selection of any boundary segments a set of files containing the ELEMents and ENURbs data will be created. A single file

```
Bxxx_m
```

where xxx are the characters (3:5) of the input file name, contains a record

```
*auto
```

and a list of the files containing the surface segment extractions. These should be reviewed to ensure the created mesh data is correct. In particular the material set number for each segment. The basic header to change is

```
ELEMent NODE=no_nd MATE=ma TYPE ....
```

The material number for the entire set can be changed by setting the value of ma desired. *It is not necessary to change the number on each element data record.* Do not edit any data in the date part ENURbs. This is used to select the correct extraction operator for each element.

The file Bxxx_m should be added to the mesh part of the input file *that created the boundary segments* as

```
include Bxxx_m
```

2.8 Problem solution

After the problem is formed, standard *FEAP* solution commands are used to solve each problem.

2.9 Graphics outputs

Standard *FEAP* plot commands may be used to display the location of the control points (PLOT NODE command) and boundary restraints on the control points (PLOT BOUNdary). Use of any contouring commands (PLOT CONT, PLOT STRE) is performed by projection onto 4-node quadrilateral sub-elements of the surface. Most plot commands may be used but there remain some delicate transformations between the various representation of the data to be plotted. For 3-d objects plots should be given in perspective mode. Thus, the data for each plot sequence should begin with

```
PLOT PERSpective  ! Required for 3-d only
PLOT HIDE         ! Required for 3-d only
```

If a problem requires long solution times it is advisable to use SAVE commands to preserve solution values prior to attempting plot outputs.

2.10 Solutions using T-splines

FEAP permits the calculation of isogeometric objects represented by *T-splines*. The solution is obtained using an *extraction operator* form in which the element shape functions are expressed in terms of shape functions given as

$$\mathbf{N}^e = \mathbf{C}^e \, \mathbf{R}^e$$

where $\mathbf{R}^e$ are a Bezier representation of NURBS, $\mathbf{C}^e$ is the element extraction operator and $\mathbf{N}^e$ are the T-spline shape functions.

The data input is provided by an output from the refinement program developed at The University of Texas by Mike Scott[27] and included as an extension of the T-Splines[28] plug-in to Rhino[29].

Only surface data is provided and thus analyses are restricted to bodies that are represented by surfaces (e.g., 2-d solid bodies, membranes and shells). A typical input file is given as:

```
FEAP * * Title information
  ndm = <2,3>     ! 2-d solids or 3-d surfaces, respectively
  ndf = <1,2,...> ! Describes number of dof at a control point

MATErial
  <SOLId, MEMBrane, SHELl>
  elastic isotropic E nu
  NURBs,TSPLine,q1,q2,q3

T-SPline
  PLOT INTErval <1 to 7>    ! Number of subdivision of surface
  FILE = "filename of data"

.... ! Loads, B.C., etc.
END               ! End of data inputs
INTEractive       ! Interactive solution commands
STOP              ! End of data file
```

Standard solution commands may be used. Graphics is available in a manner similar to that for NURBS problems.

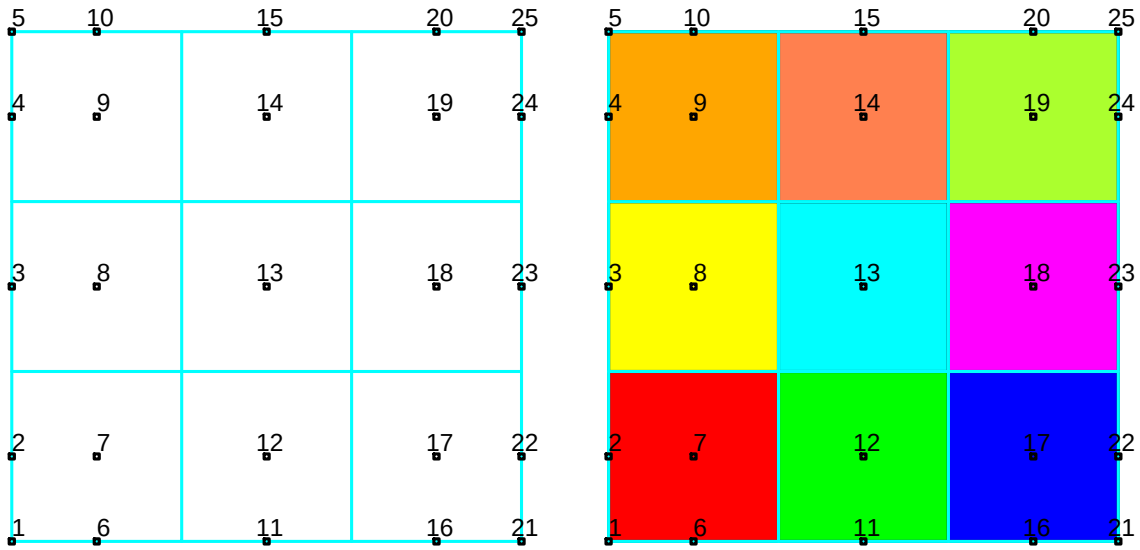
Chapter 3

Example mesh and elements

To illustrate the relationship between control points and their association with individual elements defined by knot spacing we show a few two-dimensional examples.

Example: 2-d NURBS patch of quadratic elements

The mesh and elements for a 3×3 patch of quadratic elements is shown in Fig. 3.1. The individual elements of the mesh and their associated control points are shown in Fig. 3.2.



(a) Mesh with control points (b) Elements in color

Figure 3.1: Two dimensional NURBS patch of quadratic elements.

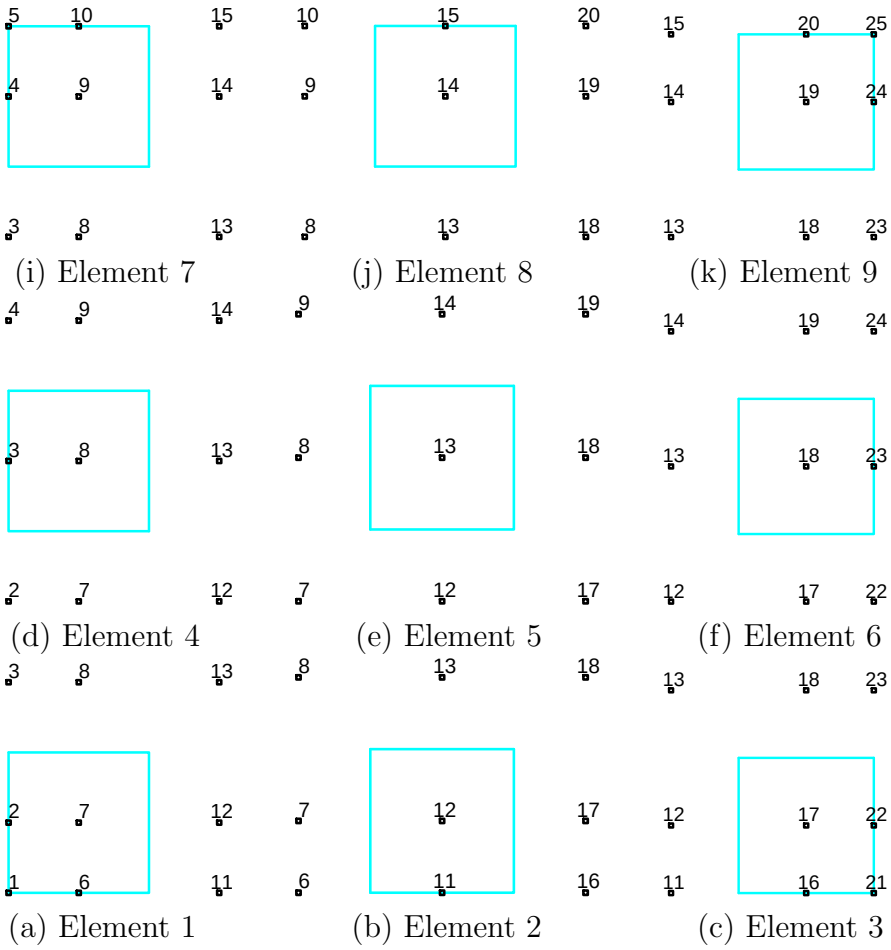


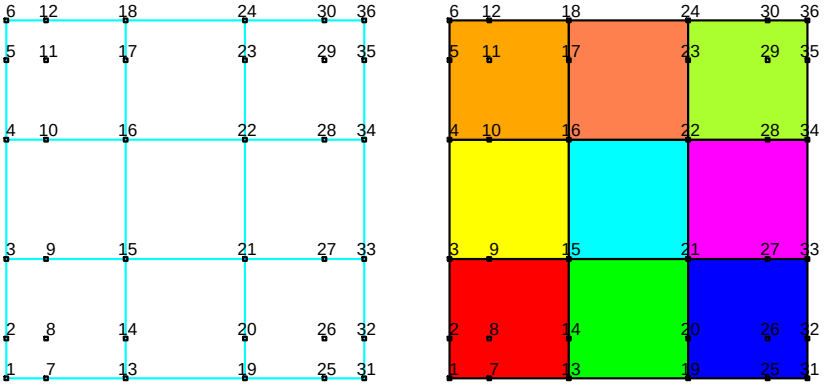
Figure 3.2: Individual elements for 2-d NURBS patch of quadratic elements.

Example: 2-d NURBS patch of cubic elements

The mesh and elements for a 3×3 patch of cubic elements is shown in Fig. 3.3. The individual elements of the mesh and their associated control points are shown in Fig. 3.4.

Example: 2-d NURBS patch of quartic elements

The mesh and elements for a 3×3 patch of quartic elements is shown in Fig. 3.5. The individual element description is not shown, however, each element involves a 5×5 patch of control point which overlap between elements except for one control point. Thus, the mesh only grows by one control point in each direction as the order is increased.



(a) Mesh with control points (b) Elements in color

Figure 3.3: Two dimensional NURBS patch of cubic elements.

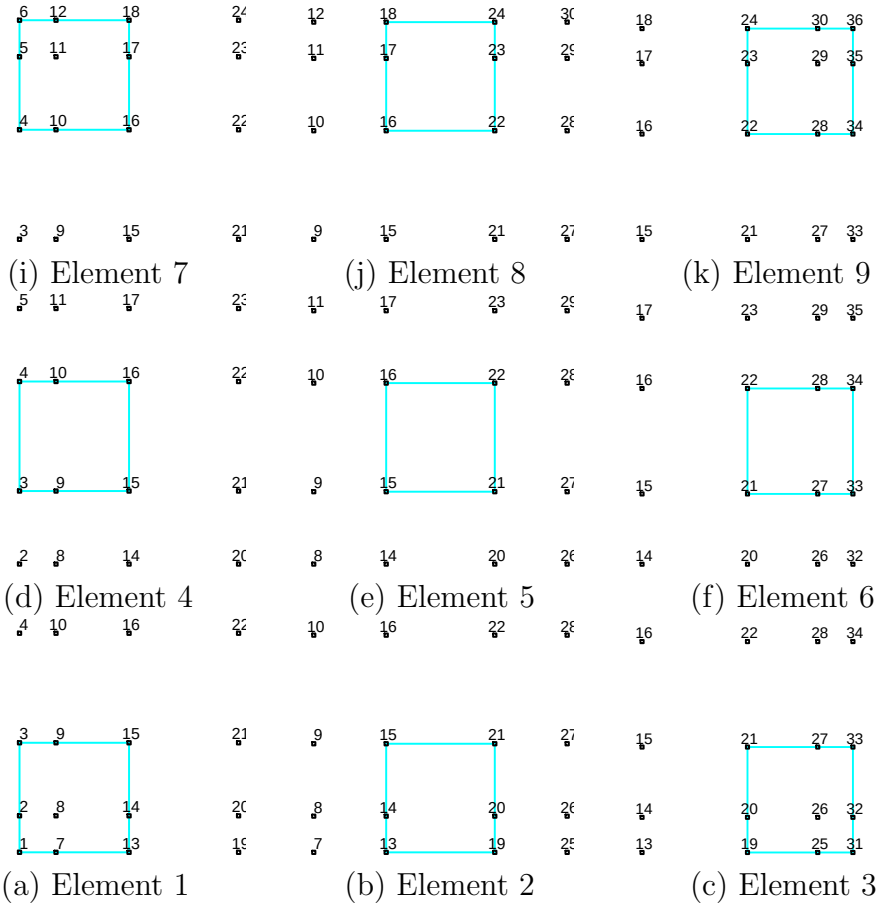
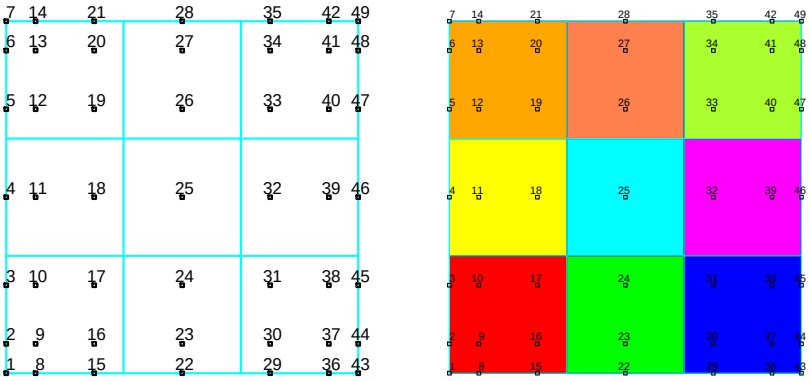


Figure 3.4: Individual elements for 2-d NURBS patch of cubic elements.



(a) Mesh with control points (b) Elements in color

Figure 3.5: Two dimensional NURBS patch of quartic elements.

Bibliography

[1] T.J.R. Hughes, J.A. Cottrell, and Y. Bazilevs. Isogeometric analysis: CAD, finite elements, NURBS, exact geometry, and mesh refinement. *Computer Methods in Applied Mechanics and Engineering*, 194:4135–4195, 2005.

[2] L. Piegl and W. Tiller. *The NURBS Book (Monographs in Visual Communication)*. Springer-Verlag, New York, 2^{nd} edition, 1997.

[3] J.A. Cottrell, T.J.R. Hughes, and Y. Bazilevs. *Isogeometric Analysis: Toward Integration of CAD and FEA*. John Wiley & Sons, New York, 2009.

[4] Y. Bazilevs, L. Beirao de Veiga, J.A. Cottrell, T.J.R. Hughes, and G. Sangalli. Isogeometric analysis: Approximation, stability and error estimates for h-refined meshes. *Mathematical Models and Methods in Applied Sciences*, 16:1031–1090, 2006.

[5] J.A. Cottrell, A. Reali, Y. Bazilevs, and T.J.R. Hughes. Isogeometric analysis of structural vibrations. *Computer Methods in Applied Mechanics and Engineering*, 195:5257–5296, 2007.

[6] Y. Bazilevs, V.M. Calo, Y. Zhang, and T.J.R. Hughes. Isogeometric fluid-structure interaction analysis with applications to arterial blood flow. *Computational Mechanics*, 38:310–322, 2006.

[7] J.A. Cottrell, T.J.R. Hughes, and A. Reali. Studies of refinement and continuity in isogeometric structural analysis. *Computer Methods in Applied Mechanics and Engineering*, 196:4160–4183, 2008.

[8] Y. Zhang, Y. Bazilevs, S. Goswami, C. Bajaj, and T.J.R. Hughes. Patient-specific vascular NURBS modeling for isogeometric analysis of blood flow. *Computer Methods in Applied Mechanics and Engineering*, 196:2943–2959, 2007.

[9] Y. Bazilevs, V.M. Calo, T.J.R. Hughes, and Y. Zhang. Isogeometric fluid-structure interaction: Theory, algorithms and computations. *Computational Mechanics*, 43:143–150, 2008.

[10] T. Elguedj, Y. Bazilevs, V.M. Calo, and T.J.R. Hughes. $\bar{B}$ and $\bar{F}$ projection methods for nearly incompressible linear and non-linear elasticity and plasticiy using higher-order NURBS elements. *Computer Methods in Applied Mechanics and Engineering*, 197:2732–2762, 2008.

[11] H. Gomez, V.M. Calo, Y. Bazilevs, and T.J.R. Hughes. Isogeometric analysis of the Cahn-Hilliard phase-field model. *Computer Methods in Applied Mechanics and Engineering*, 197:4333–4352, 2008.

[12] T.J.R. Hughes, A. Reali, and G. Sangalli. Duality and unified analysis of discrete approximations in structural dynamics and wave propagation: Comparison of p-method finite elements with k-method NURBS. *Computer Methods in Applied Mechanics and Engineering*, 197:4104–4124, 2008.

[13] W.A.. Wall, M.A. Frenzel, and C. Cyron. Isogeometric structural shape optimization. *Computer Methods in Applied Mechanics and Engineering*, 197:2976–2988, 2008.

[14] J.A. Evans, Y. Bazilevs, I. Babuška, and T.J.R. Hughes. n-widths, sup-infs, and optimality ratios for the k-version of the isogeometric finite element method. *Computer Methods in Applied Mechanics and Engineering*, 198:1726–1741, 2009.

[15] J. Lu. Circular element: Isogeometric elements of smooth boundary. *Computer Methods in Applied Mechanics and Engineering*, 198:2391–2402, 2009.

[16] J. Kiendl, K.-U. Bletzinger, J. Linhard, and R. Wüchner. Isogeometric shell analysis with Kirchhoff-Love elements. *Computer Methods in Applied Mechanics and Engineering*, 198:3902–3914, 2009.

[17] Josef M. Kiendl. *Isogeometric Analysis and Shape Optimal Design of Shell Structures*. Doktor-ingenieurs dissertation, Lehrstuhl für Statik, Technische Universität München, Munich, Germany, 2010.

[18] Y. Bazilevs, V.M. Calo, J.A. Cottrell, J.A. Evans, T.J.R. Hughes, S. Lipton, M.A. Scott, and T.W. Sederberg. Isogeometric analysis using T-splines. *Computer Methods in Applied Mechanics and Engineering*, 199:229–263, 2010.

[19] R. Echter and M. Bischoff. Numerical efficiency, locking and unlocking of NURBS finite elements. *Computer Methods in Applied Mechanics and Engineering*, 199:374–382, 2010.

[20] D.J. Benson, Y. Bazilevs, M.C. Hsu, and T.J.R. Hughes. Isogeometric shell analysis: The Reissner-Mindlin shell. *Computer Methods in Applied Mechanics and Engineering*, 199:276–289, 2010.

[21] D.J. Benson, Y. Bazilevs, E. De Luycker, M.-C. Hsu, M. Scott, T.J.R. Hughes, and T. Belytschko. A generalized finite element formulation for arbitrary basis functions: From isogeometric analysis to XFEM. *International Journal for Numerical Methods in Engineering*, 83:765–785, 2010.

[22] D.J. Benson, Y. Bazilevs, M.-C. Hsu, and T.J.R. Hughes. A large deformation, rotation-free, isogeometric shell. *Computer Methods in Applied Mechanics and Engineering*, 200:1367–1378, 2011.

[23] R.L. Taylor. Isogeometric analysis of nearly incompressible solids. *International Journal for Numerical Methods in Engineering*, 87(1–5):273–288, 2011.

[24] R.L. Taylor. *FEAP - A Finite Element Analysis Program, User Manual*. University of California, Berkeley. `projects.ce.berkeley.edu/feap`.

[25] O.C. Zienkiewicz, R.L. Taylor, and J.Z. Zhu. *The Finite Element Method: Its Basis and Fundamentals*. Elsevier, Oxford, 7^{th} edition, 2013.

[26] M.J. Borden, M.A. Scott, J.A. Evans, and T.J.R. Hughes. Isogeometric finite element data structures based on Bézier extraction of NURBS. *International Journal for Numerical Methods in Engineering*, 87(1–5):15–47, 2011.

[27] M.A. Scott, M.J. Borden, C.V. Verhoosel, T.W. Sederberg, and T.J.R. Hughes. Isogeometric finite element data structures based on Bézier extraction of T-splines. Technical Report ICES Report 10-45, The Institute for Computational Engineering and Sciences, The University of Texas at Austin, November 2010.

[28] T-Splines, Inc. http://www.tsplines.com/products/tsplines-for-rhino.html.

[29] Rhinoceros: NURBS modeling for Windows. http://www.rhino3d.com.

Appendix A

BLOCk input form

A.1 NSIDE list of control points

The side nodes for a patch of NURBS is a list of control point numbers. The data is given by

```
NSIDes
  side  s1 lside1 k1 (cp1(i),i=1,lside1)
  side  s2 lside2 k2 (cp2(i),i=1,lside2)
    .....
    ! terminate with blank record
```

where `s1` is the side number, `lside1` is the number of control point numbers in the side vector, `k1` is the number of the knot vector associated with the side; and `cp1(i)` is the list of `lside1` control point numbers defining the side. The number of control points and the number of associated knot values is related by

```
lknot(k1) = lside(s1) + order(k1) + 1
```

Thus, a linear side (order = 1) with two (2) control points has a knot vector with four (4) values. The open knot vector has repeated first and last values and thus is the form

```
0.0  0.0  1.0  1.0
```

or similar.

A.2 NBLOCK specification

A.2.1 One dimensional block

For a one dimensional block the command is given as

```
NBLOck
  block 1 ma side
```

where `ma` is the block material set number and `side` the side number giving the list of control points.

A.2.2 Two dimensional block

A two dimensional block is described by two knot vectors. The first knot vector describes the two sides of the block and also any intermediate list of sides in the interior of the block directed in the same direction. The second knot vector describes the second direction in the block. The associated control points for the second knot vector must be the first control point from all the sides comprising the first direction and given in the order corresponding to increasing knot values in the second direction. The input data for the block is given as

```
NBLOck
  block 2 ma side2
```

where `ma` the material set number and `side2` the side number for the second block direction. For example, the block shown in Fig. A.1 can let block direction 1 be

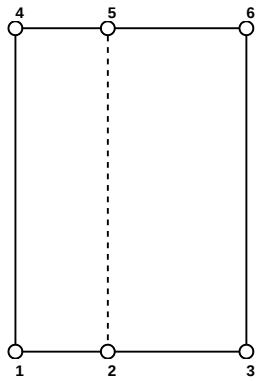


Figure A.1: Two dimensional NURBS block data

associated with side with control points 1 and 4 and block direction 2 with control points 1, 2, and 3.

Example: 2-d NURB BLOCK

The data for the two dimensional tensor product block shown in Figure A.1 is specified as follows:

```
NURBs
  1  0   0.0   0.0  1.0
  2  0  40.0   0.0  1.0
  3  0 100.0   0.0  1.0
  4  0   0.0 140.0  1.0
  5  0  40.0 140.0  1.0
  6  0 100.0 140.0  1.0

KNOTs
  open 1  4  0.0 0.0 1.0 1.0
  open 2  5  0.0 0.0 0.5 1.0 1.0

NSIDEs
  side 1  2  1  1  4
  side 2  2  1  2  5
  side 3  2  1  3  6
  side 4  3  2  1  2  3

NBLOCK
  block  2  1  4
```

Note that the specification of the dotted line at the knot value 0.5 is not necessary to give an exact geometry for this simple rectangle. However, it permits for a non-uniform subdivision in the horizontal direction later.

A.2.3 Three dimensional block

The specification of a three dimensional NURBS block is given by defining the list of all the sides lists for one of the block directions. In Fig. A.2 the sides in the 3 knot-direction are used to define a rectangular block. The NURBS block command for a three dimensional block is given as

```
NBLOCK
  block 3 ma k1 k2
  (side(i),i = 1,list3d)
```

where `ma` is the material set number; `k1, k2` are the knot numbers defining the generator plane of the block; and `side(i)` is the list of sides perpendicular to the generator plane.

Example: Three dimensional rectangular block

The data for the rectangular block shown in Fig. A.2 is given by the control point locations for nodes 1 to 12 and the following knot vectors; side lists; and block command:

```
KNOTS
  open 1  6  0.0 0.0 0.0 1.0 1.0 1.0
  open 2  4  0.0 0.0 1.0 1.0
  open 3  4  0.0 0.0 1.0 1.0

NSIDES
  side 1  2  3   1  2
  side 2  2  3   3  4
  side 3  2  3   5  6
  side 4  2  3   7  8
  side 5  2  3   9 10
  side 6  2  3  11 12

NBLOCK
  block 3 1  1  2
    1 2 3 4 5 6
```

The direction 1 of the block is a quadratic NURBS while directions 2 and 3 are linear NURBS. Since the 1-direction is quadratic, the first 3 side lists will be used to define this direction while the second set of three will create the linear behavior for the 2-direction.

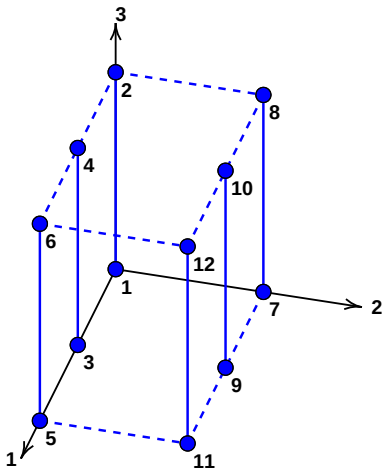


Figure A.2: Three dimensional NURBS block data

Appendix B

Patch storage arrays

The definition of NURBS patches utilizes several arrays to facilitate the description of shape, element definition and knot intervals. The basic description is given in the following tables.

NAME	`line`	`surf`	`soli`	Description
`NBLK(1,ib)`	1	2	3	Dimension of patch
`NBLK(2,ib)`	ma	ma	ma	Material number
`NBLK(3,ib)`	nreg	nreg	nreg	Region number
`NBLK(4,ib)`	pn1	pn2	pn1*pn2	Length of patch
`NBLK(5,ib)`	ipart	ipart	ipart	Part number
`NBLK(6,ib)`	is	pn2+1	k1	Knot 1
`NBLK(7,ib)`	0	0	k2	Knot 2
`NBLK(8,ib)`	-	eside(1)	k3	Edge 1/Knot 3
`NBLK(9,ib)`	-	eside(2)	p1	Edge 2/Length 1
`NBLK(10,ib)`	-	eside(3)	p2	Edge 3/Length 2
`NBLK(11,ib)`	-	eside(4)	p3	Edge 4/Length 3

Table B.1: NURBS patch array `NBLK` parameters for block `ib`.

NAME	line	surf	soli	Description
NBLKSD(1,ib)	is	is	is	Side 1 number
to				
NBLKSD(L,ib)	-	l=pn2	l=pn1*pn2	Side L number from NBLK

Table B.2: NURBS patch array NBLKSD parameters for block ib.

NAME	Description
LSIDE(1,is)	Number of control points.
LSIDE(2,is)	Knot number

Table B.3: NURBS patch array LSIDE parameters for side is.

NAME	Description
NSIDES(1,is)	Control point 1.
NSIDES(2,is)	Control point 2.
etc.	
NSIDES(dsideig,is)	Last control point.

Table B.4: NURBS patch array NSIDES parameters for side is.

NAME	Description
LKNOT(0,kn)	Knot type: Open = 1;
LKNOT(1,kn)	Number of knots.
LKNOT(2,kn)	Order of knots.
LKNOT(3,kn)	Number control points.
LKNOT(4,kn)	Used to elevate order.

Table B.5: NURBS knot array LKNOT for knot kn.