Refinement of B-Splines and NURBS:

One of the most interesting aspects of B-splines is the variety of ways in which the basis may be enriched while leaving the underlying geometry and its parametrization unchanged. To fully recognize the many possibilities, we will need to understand the basic mechanisms of B-spline refinement and how they differ from classical finite element analysis. In particular, not only do we have control over the element size and the order of the basis, but we can control the continuity of the basis as well.

Knot Insertion:

The first mechanism by which one can enrich the basis is knot insertion. There are actually two different views of knot insertion, the analysis point of view and the geometry point of view.

Basis Functions ←

Analysis Point of View: Knot insertion is the process of enriching a B-spline basis by inserting a new knot into the existing knot vector.

Control Points ←

Geometry Point of View: Knot insertion is the process of refining a B-spline curve, surface, or solid without changing its shape or parametrization by inserting a new knot into the knot vector.

The connection between these two points of view is that refinement of a B-spline geometry is accomplished by performing knot insertion on the underlying B-spline basis and computing new control points.

Mathematically, suppose that we are given a B-spline curve:

$$\vec{C}(\xi) = \sum_{i=1}^{n} \vec{P}_i \, N_{i,p}(\xi)$$

and an associated polynomial degree p and knot vector:

$$\Xi = \{\xi_1, \xi_2, \ldots, \xi_{n+p+1}\}$$

The process of knot insertion is akin to inserting some knot $t \in (\xi_1, \xi_{n+p+1})$ into Ξ, resulting in the extended knot vector:

$$\bar{\Xi} = \{\bar{\xi}_1 = \xi_1, \bar{\xi}_2 = \xi_2, \ldots, \bar{\xi}_k = \xi_k, \bar{\xi}_{k+1} = t, \bar{\xi}_{k+2} = \xi_{k+1}, \ldots, \bar{\xi}_{m+p+1} = \xi_{n+p+1}\}$$

where $m = n+1$. We then have a new set of basis functions $\{\bar{N}_{j,p}\}_{j=1}^{m}$ associated with $\bar{\Xi}$.

From an analysis point of view, the primary concern of the knot insertion process is finding some relationship between the original basis $\{N_{i,p}\}_{i=1}^{n}$ and the new basis $\{\bar{N}_{j,p}\}_{j=1}^{m}$.

From a geometry point of view, the primary concern of the knot insertion process is finding a new set of control points $\{\vec{Q}_j\}_{j=1}^{m}$ such that:

$$\vec{C}(\xi) = \sum_{i=1}^{n} \vec{P}_i \, N_{i,p}(\xi) = \sum_{j=1}^{m} \vec{Q}_j \, \bar{N}_{j,p}(\xi)$$

However, note that if we have a relationship of the form:

$$N_{i,p}(\xi) = \sum_{j=1}^{m} T_{ji}^{p}\, \bar{N}_{j,p}(\xi) \qquad (*)$$

Then we have:

$$\begin{aligned}
\vec{C}(\xi) &= \sum_{i=1}^{n} \vec{P}_i\, N_{i,p}(\xi) \\
&= \sum_{i=1}^{n} \vec{P}_i \left(\sum_{j=1}^{m} T_{ji}^{p}\, \bar{N}_{j,p}(\xi) \right) \\
&= \sum_{j=1}^{m} \left(\sum_{i=1}^{n} T_{ji}^{p}\, \vec{P}_i \right) \bar{N}_{j,p}(\xi) \\
&= \sum_{j=1}^{m} \vec{Q}_j\, \bar{N}_{j,p}(\xi)
\end{aligned}$$

where:

$$\vec{Q}_j = \sum_{i=1}^{n} T_{ji}^{p}\, \vec{P}_i \qquad (**)$$

Consequently, we are able to address the concern of geometry by first addressing the concern of analysis (or vice versa).

Let us attempt to tackle the problem of finding an expression in the form of (*). Note first that:

$$\bar{N}_{i,p}(\xi) = N_{i,p}(\xi) \quad \text{for } i = 1, \ldots, k-p-1$$

$$\bar{N}_{i,p}(\xi) = N_{i-1,p}(\xi) \quad \text{for } i = k+2, \ldots, m$$

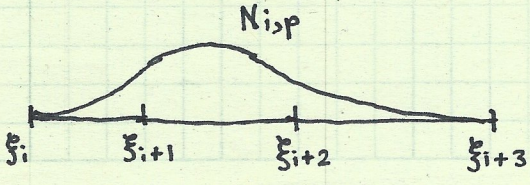
or, identically:

$$N_{i,p}(\xi) = \begin{cases} \bar{N}_{i,p}(\xi) & \text{for } i = 1, \ldots, k-p-1 \\ \bar{N}_{i+1,p}(\xi) & \text{for } i = k+1, \ldots, n \end{cases}$$

So what about $N_{i,p}(\xi)$ for $i = k-p, \ldots, k$? We may interpret $N_{i,p}(\xi)$ as being the single B-spline basis function associated with the local knot vector:

$$\underline{\Xi}_{loc,i} = \{ \xi_i, \xi_{i+1}, \ldots, \xi_{i+p+1} \}$$

Visually:

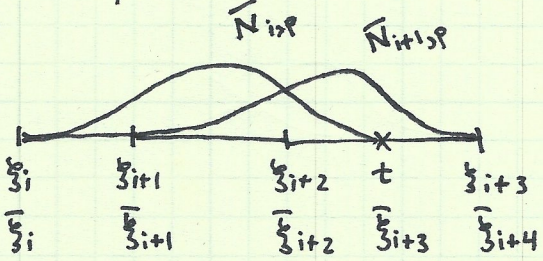


$p = 2$

Now suppose that we insert t into $\bar{\Xi}_{loc,i}$, resulting in the extended local knot vector:

$$\bar{\Xi}_{loc,i} = \{ \bar{\xi}_i = \xi_i, \ldots, \bar{\xi}_\ell = t, \ldots, \bar{\xi}_{i+p+2} = \xi_{i+p+1} \}$$

Then there are two basis functions associated with $\bar{\Xi}_{loc,i}$, and they are precisely $\bar{N}_{i,p}(\xi)$ and $\bar{N}_{i+1,p}(\xi)$. Visually:



We thus expect $N_{i,p}(\xi)$ to be a linear combination of $\bar{N}_{i,p}(\xi)$ and $\bar{N}_{i+1,p}(\xi)$. Indeed, using recursion and Cox-de Boor, it can be shown that:

$$N_{i,p}(\xi) = \left(\frac{t - \bar{\xi}_i}{\bar{\xi}_{i+p+1} - \bar{\xi}_i} \right) \bar{N}_{i,p}(\xi) + \left(\frac{\bar{\xi}_{i+p+2} - t}{\bar{\xi}_{i+p+2} - \bar{\xi}_{i+1}} \right) \bar{N}_{i+1,p}(\xi)$$

or, in composite form:

Basis Enrichment

$$N_{i,p}(\xi) = \begin{cases} \bar{N}_{i,p}(\xi) & \text{for } i = 1, \ldots, k-p-1 \\ \left(\frac{t - \bar{\xi}_i}{\bar{\xi}_{i+p+1} - \bar{\xi}_i} \right) \bar{N}_{i,p}(\xi) + \left(\frac{\bar{\xi}_{i+p+2} - t}{\bar{\xi}_{i+p+2} - \bar{\xi}_{i+1}} \right) \bar{N}_{i+1,p}(\xi) & \text{for } i = k-p, \ldots, k \\ \bar{N}_{i+1,p}(\xi) & \text{for } i = k+1, \ldots, n \end{cases}$$

Recognizing the above as our desired expression for (*), we can transpose the relationship to find our desired expression for (**):

Curve Refinement

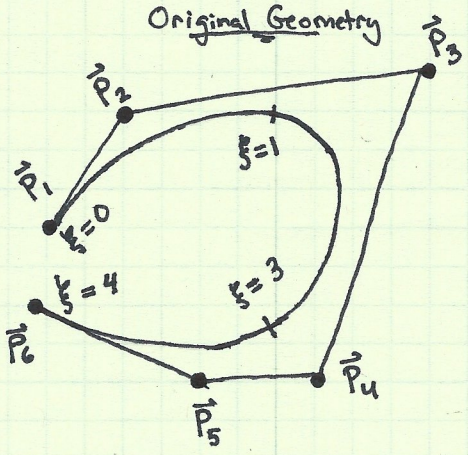
$$\vec{Q}_j = \begin{cases} \vec{P}_1 & \text{for } j = 1 \\ \alpha_j \vec{P}_j + (1 - \alpha_j) \vec{P}_{j-1} & \text{for } j = 2, \ldots, m-1 \\ \vec{P}_n & \text{for } j = m \end{cases}$$

where:

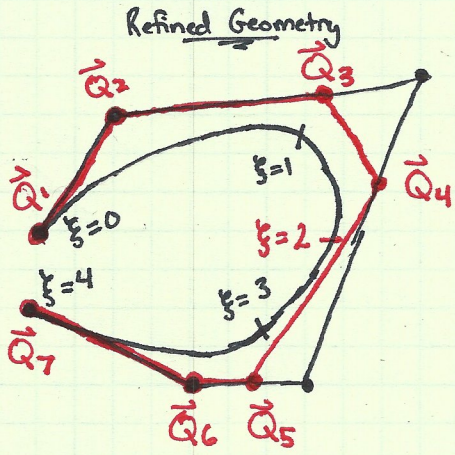
$$\alpha_j = \begin{cases} 1 & \text{for } j = 1, \ldots, k-p \\ \left(\frac{t - \xi_j}{\xi_{j+p} - \xi_j} \right) & \text{for } j = k-p+1, \ldots, k \\ 0 & \text{for } j = k+1, \ldots, m \end{cases}$$

The above refinement algorithm is referred to as Boehm's algorithm. In it, new control points are computed as convex combinations of old control points, and hence it is very stable. It is demonstrated visually below.

Original Geometry



Refined Geometry



Polynomial Degree: $p = 3$

Knot Vector: $\Xi = \{0, 0, 0, 0, 1, 3, 4, 4, 4, 4\}$

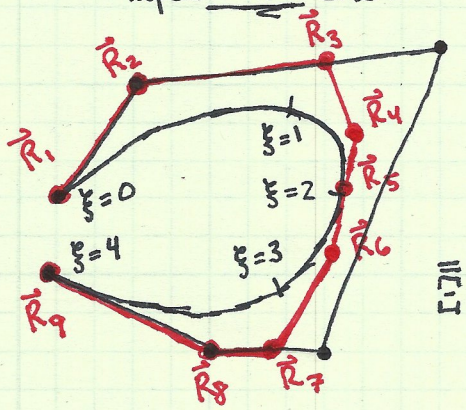
Polynomial Degree: $p = 3$

Knot Vector: $\bar{\Xi} = \{0, 0, 0, 0, 1, 2, 3, 4, 4, 4, 4\}$

Note that the new control polygon lies closer to the curve. In fact, if we repeat the knot insertion process enough times such that a given knot has multiplicity p*, then the control polygon that results will touch the curve. Indeed, this is precisely what de Boor's recursive evaluation routine executes.

* or $p+1$!

Repeated Knot Insertion



Polynomial Degree: $p = 3$

Knot Vector:

$$\bar{\bar{\Xi}} = \{0, 0, 0, 0, 1, \underbrace{2, 2, 2}_{\text{Repeated } p \text{ times!}}, 3, 4, 4, 4, 4\}$$

Often times, we are not only interested in just adding a single knot into a knot vector but rather several. In the design community, this is referred to as knot refinement, but we will also refer to it as knot insertion. Given a degree p and knot vector Ξ, we introduce an extended knot vector that contains all of the knots we desire to add:

$$\bar{\Xi} = \{\bar{\xi}_1 = \xi_1, \bar{\xi}_2, \ldots, \bar{\xi}_{m+p+1} = \xi_{n+p+1}\}$$

Where:

$$m = n + \text{Number of Added Knots}$$

The original basis is denoted as $\{N_{i,p}\}_{i=1}^{n}$ while the refined basis is denoted as $\{\bar{N}_{j,p}\}_{j=1}^{m}$. Likewise, the original control polygon is denoted as $\{\vec{P}_i\}_{i=1}^{n}$ while the new refined control polygon is denoted as $\{\vec{Q}_j\}_{j=1}^{m}$.

In 1980, Cohen, Lyche, and Riesenfeld showed that:

$$N_{i,p}(\xi) = \sum_{j=1}^{m} T_{ji}^{p}\, \bar{N}_{j,p}(\xi) \qquad i = 1,\dots,n$$

$$\vec{Q}_j = \sum_{i=1}^{n} T_{ji}^{p}\, \vec{P}_i \qquad j = 1,\dots,m$$

where the refinement matrix T_{ji}^{p} is built recursively:

$$T_{ji}^{0} = \begin{cases} 1 & \text{if } \bar{\xi}_j \in [\xi_i, \xi_{i+1}) \\ 0 & \text{otherwise} \end{cases}$$

$$T_{ji}^{k} = \left(\frac{\bar{\xi}_{j+k} - \xi_i}{\xi_{i+k} - \xi_i}\right) T_{ji}^{k-1} + \left(\frac{\xi_{i+k+1} - \bar{\xi}_{j+k}}{\xi_{i+k+1} - \xi_{i+1}}\right) T_{j,i+1}^{k-1}$$

$$\text{for } k = 1,\dots,p$$

The above forms the basis of the Oslo algorithm. Notably, repeated substitution of the above recursive definition and re-indexing leads to the following:

$$\vec{Q}_j = \sum_{i=1}^{n+q-1} \vec{P}_{q,ij}\, T_{ji}^{p+1-q} \qquad q = 1,\dots,p+1$$

where:

$$\vec{P}_{q,ij} = (1 - \alpha_{q,ij})\, \vec{P}_{q-1,i-1,j} + \alpha_{q,ij}\, \vec{P}_{q-1,ij}$$

$$q = 2,\dots,p+1$$
$$i = 1,\dots,n+q-1$$

and

$$\vec{P}_{1,ij} = \vec{P}_i$$

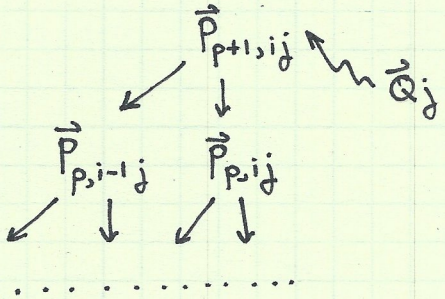
with:

$$\alpha_{q,ij} = \frac{\bar{\xi}_{j+q} - \xi_i}{\xi_{i+q} - \xi_i}$$

Note immediately that:

$$\vec{Q}_j = \vec{P}_{p+1,ij} \qquad \text{where } \bar{\xi}_j \in [\xi_i, \xi_{i+1})$$

and we can thus compute $\vec{Q}_j$ using the triangular scheme:



This inspires the following pseudocode:

```
function Refined Control Point (Q_j)
begin
        find i s.t. ξ̄_j ∈ [ξ_i, ξ_{i+1}) via binary search
        return Oslo(p+1, i, j)
end
```

```
function Oslo(q, i, j)
begin
        if q=1 then
            return P_i
        else
            compute α_{q,ij} = (ξ̄_{j+q-1} - ξ_i)/(ξ_{i+q-1} - ξ_i)
            return (1 - α_{q,ij}) * Oslo(q-1, i-1, j)
                   +  α_{q,ij} * Oslo(q-1, i, j)
end
```

The Oslo algorithm suffers from several inefficiencies which have been addressed in the literature, but these are beyond the scope of this class. Moreover, there are many other ways in which one can perform knot insertion. A promising method in the context of isogeometric analysis is Bézier projection, the subject of the recent paper:

> D. Thomas, M.A. Scott, J.A. Evans, K. Tew, and E.J. Evans, "Bézier projection: A unified approach for local projection and quadrature-free refinement and coarsening of NURBS and T-splines with particular application to isogeometric design and analysis," accepted for publication in CMAME.

Bézier projection is built upon the concept of Bézier extraction, which is the backbone of modern implementations of isogeometric analysis.

Note that we may recast knot insertion in terms of matrix equations. Defining:

$$\underline{N} = \begin{bmatrix} N_{1,p} \\ \vdots \\ N_{n,p} \end{bmatrix}, \quad \underline{\bar{N}} = \begin{bmatrix} \bar{N}_{1,p} \\ \vdots \\ \bar{N}_{m,p} \end{bmatrix}, \quad \underline{P}_A = \begin{bmatrix} (\vec{P}_1)_A \\ \vdots \\ (\vec{P}_n)_A \end{bmatrix}, \quad \underline{Q}_A = \begin{bmatrix} (\vec{Q}_1)_A \\ \vdots \\ (\vec{Q}_m)_A \end{bmatrix}$$

$A = 1, \ldots, d$

$$\underline{\underline{T}} = \begin{bmatrix} T_{11}^p & \cdots & T_{1n}^p \\ \vdots & \ddots & \vdots \\ T_{m1}^p & \cdots & T_{mn}^p \end{bmatrix}$$

we have:

$$\underline{N} = \underline{\underline{T}}^T \underline{\bar{N}} \quad \text{and} \quad \underline{Q}_A = \underline{\underline{T}}\, \underline{P}_A \quad A = 1, \ldots, d$$

In multiple dimensions, we have:

$$\underline{N} = \underline{N}^{(1)} \otimes \underline{N}^{(2)} \otimes \cdots \otimes \underline{N}^{(d_s)}$$

where:

$$\underline{N} = \begin{bmatrix} N_{(1,\ldots,1),(p_1,\ldots,p_{d_s})} \\ \vdots \\ N_{(n_1,\ldots,n_{d_s}),(p_1,\ldots,p_{d_s})} \end{bmatrix} \qquad \underline{N}^{(j)} = \begin{bmatrix} N_{1,p_j}^{(j)} \\ \vdots \\ N_{n_j,p_j}^{(j)} \end{bmatrix}$$

$$\underline{a} \otimes \underline{b} = \begin{bmatrix} a_1 \underline{b} \\ \vdots \\ a_{\mathcal{N}} \underline{b} \end{bmatrix} \qquad \mathcal{N} = \text{no. of rows of } A$$

Consequently, if we perform knot insertion in each direction:

$$\underline{N}^{(j)} = \left(\underline{\underline{T}}^{(j)}\right)^T \underline{\bar{N}}^{(j)} \qquad \text{for } j = 1, \ldots, d_s$$

then we immediately see that multi-dimensional knot insertion is performed simply through tensor products:

$$\underline{N} = \underline{\underline{T}}^T \underline{\bar{N}} \quad \text{and} \quad \underline{Q}_A = \underline{\underline{T}}\, \underline{P}_A \quad A = 1, \ldots, d$$

where:

$$\underline{\underline{T}} = \underline{\underline{T}}^{(1)} \otimes \underline{\underline{T}}^{(2)} \otimes \cdots \otimes \underline{\underline{T}}^{(d_s)}$$

and the operator $\otimes$ is defined for two matrices as:

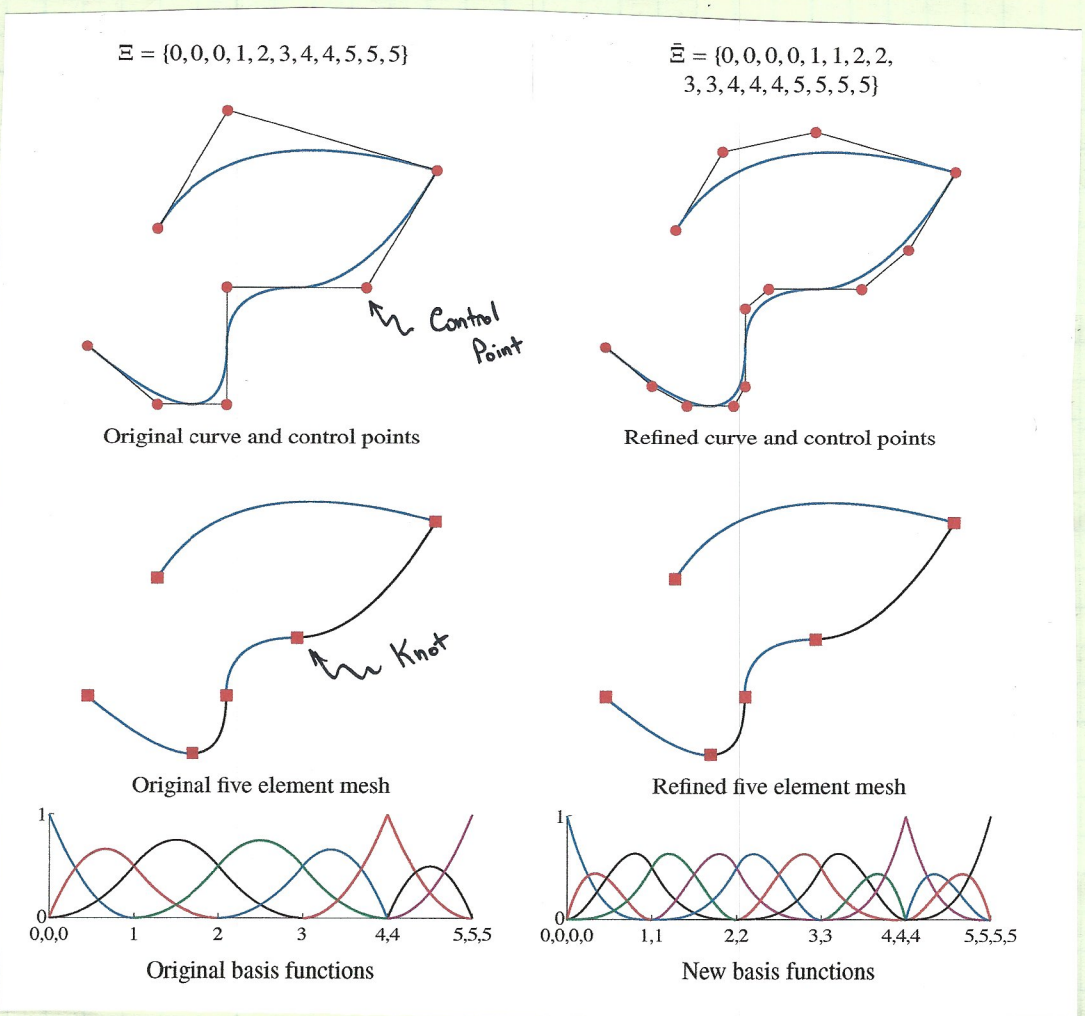
$$\underline{\underline{A}} \otimes \underline{\underline{B}} = \begin{bmatrix} A_{11}\underline{\underline{B}} & \cdots & A_{1M}\underline{\underline{B}} \\ \vdots & \ddots & \vdots \\ A_{N1}\underline{\underline{B}} & \cdots & A_{NM}\underline{\underline{B}} \end{bmatrix}$$

N = no. of rows of $\underline{\underline{A}}$
M = no. of columns of $\underline{\underline{A}}$

Knot insertion of NURBS geometries is performed by simply applying knot insertion to the projective B-spline geometry.

Degree Elevation:

The second mechanism by which one can enrich the basis is degree elevation. As its name implies, the process involves raising the polynomial degree of the basis functions used to represent the geometry. As the original basis has $p - m_i$ continuous derivatives across knot ξ_i where m_i is the multiplicity of ξ_i, it is clear that when p is increased, m_i must be increased as well to preserve the discontinuities in the various derivatives existing in the original curve. Indeed, during degree elevation, the multiplicity of each knot value is increased by one, but no new knot values are added. As with knot insertion, neither the geometry nor the parametrization are changed. We have illustrated degree elevation in the setting of a elevating a quadratic B-spline curve.



Quadratic B-spline

Cubic B-spline

The process of degree elevation begins by replicating existing knots until their multiplicity is equal to the polynomial degree, thus effectively subdividing the curve into many Bézier curves by knot insertion. This process is referred to as Bézier extraction. The next step is to elevate the order or degree of the polynomial on each of these segments. Finally, excess knots are removed by knot deletion to combine the segments into one, degree-elevated B-spline curve. There are many different approaches to knot deletion, which effectively coarsens the B-spline basis. In the context of isogeometric analysis, Bézier projection is likely the most straightforward approach.

Like knot insertion, multi-dimensional degree elevation is performed through tensor products, and degree elevation of NURBS geometries is performed by simply applying degree elevation to the projective B-spline geometry.

Relationship with Classical Finite Element Strategies:

Both knot insertion and degree elevation are closely related to classical finite element refinement strategies. Recall that in classical finite elements, the two approaches to basis refinement are h-refinement and p-refinement. During h-refinement, individual finite elements are subdivided, which is analogous to subdividing knot intervals through knot insertion. During p-refinement, the basis functions on individual finite elements are degree elevated, which is analogous to degree elevating a given B-spline. In contrast with classical finite elements, we have a third notion of refining basis functions in the context of B-splines. Namely, we can modify the continuity of a B-spline basis by replicating knots via knot insertion. We refer to this process as basis roughening, or in the lingo of finite elements, k-refinement. These different strategies are outlined below.

Refinement Strategy	Meaning	Operation
h-refinement	Cell subdivision	Knot insertion
p-refinement	Degree elevation	Degree elevation
k-refinement	Basis roughening	Knot insertion

These different strategies may be combined, resulting in a general hpk-refinement method, analogous to hp-methods in finite element analysis. We may also coarsen a basis, resulting in the following strategies.

Refinement Strategy	Meaning	Operation
h-coarsening	Cell merging	Knot deletion
p-coarsening	Degree reduction	Degree reduction
k-coarsening	Basis smoothing	Knot deletion

It should be mentioned that coarsening operations are not unique nor are they obvious. However, a discussion of coarsening is beyond the scope of this class. For a further discussion, see:

D. Thomas, M.A. Scott, J.A. Evans, K. Tew, and E.J. Evans, "Bézier projection: A unified approach for local projection and quadrature-free refinement of NURBS and T-splines...", accepted for publication in CMAME.