Adaptivity with B-spline Elements

Malcolm Sabin(✉)

Numerical Geometry Ltd., 19 John Amner Close, Ely, Cambs, UK
malcolm.sabin@btinternet.com

Abstract. This paper takes a stage further the work of Kraft [1] and of Grinspun et al. [2] who used subdivision formulations to show that finite element formulation can be expressed better in terms of the basis functions used to span the space, rather than in terms of the partitioning of the domain into elements. Adaptivity is achieved not by subpartitioning the domain, but by nesting of solution spaces. This paper shows how, with B-spline elements, their approach can be further simplified: a B-spline element of any degree and in any number of dimensions can be refined independently of every other within the basis. This completely avoids the linear dependence problem, and can also give slightly more focussed adaptivity, adding extra freedom only, and exactly, where it is needed, thus reducing the solution times.

Keywords: Finite elements · Adaptivity · Nested spaces

1 Motivation

In the use of the finite element method to solve partial differential equations such as those for thermal analysis and elasticity, an important tool for achieving the best trade-off between accuracy and computing cost is the use of *adaptivity*, whereby the resolution of an initial analysis is improved by using a finer mesh where the solution is varying rapidly, but only there. In places where the solution is already captured to an adequate accuracy, adding extra freedoms adds significant extra cost to the solution without contributing to reducing its overall inaccuracy. The use of B-spline functions as a basis for the analysis turns out to provide a very good way of making an optimal trade-off, which has none of the disadvantages of local remeshing. The key to this is thinking in terms of the basis functions instead of the partitioning of the domain.

2 Prior Work

2.1 Kraft

In [1] Kraft described how a local increase of density in an array of quadratic B-spline functions could be achieved by replacing a single bi-quadratic B-spline function by 16 others. The following Figs. 1, 2, 3 and 4 are my transcriptions of those in his paper.

M. Floater et al. (Eds.): MMCS 2016, LNCS 10521, pp. 218–232, 2017.
https://doi.org/10.1007/978-3-319-67885-6_12

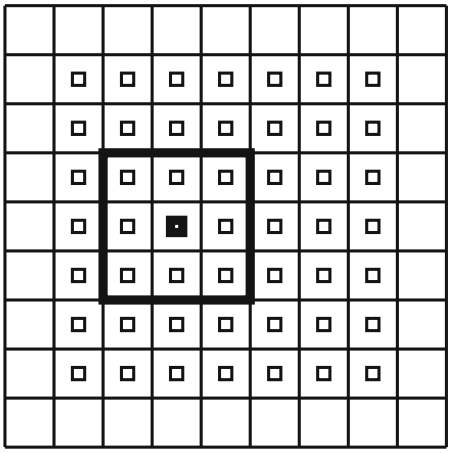

Fig. 1. In this figure the regular grid lines denote knot lines. The small squares denote the centres of biquadratic B-spline functions over the domain in the plane of the paper. The small black square denotes the centre of a specific such function with the boundary of its support highlight. Kraft takes such a function as a candidate for adaptivity.

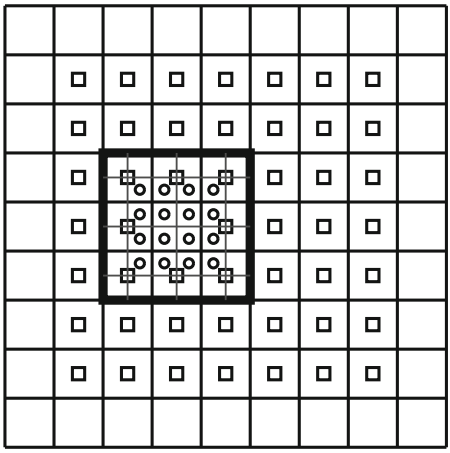

Fig. 2. In this figure the function denoted by the black square of Fig. 1 is replaced by the sixteen functions denoted here by the small circles. Those sixteen functions have knot lines which include the extra lines inside the support of the original function. Note that the remaining functions whose centres are at the squares do not see these extra knots. This process can be viewed as having the new functions defined by knot insertion into the old one (just the old one: no others) or as having the new knots appear as a side-effect of the replacement of the old function by the new ones.

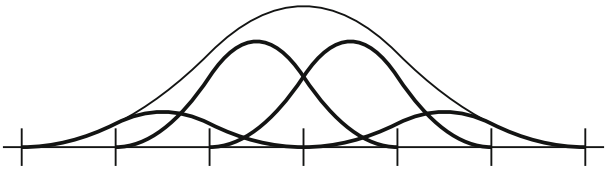

Fig. 3. The new functions are defined by the tensor product of two univariate refinements. In each the replacement is of the upper function here by the four lower ones. Because the new functions sum to the old one the partition of unity property is maintained. The total number of new functions is 16.

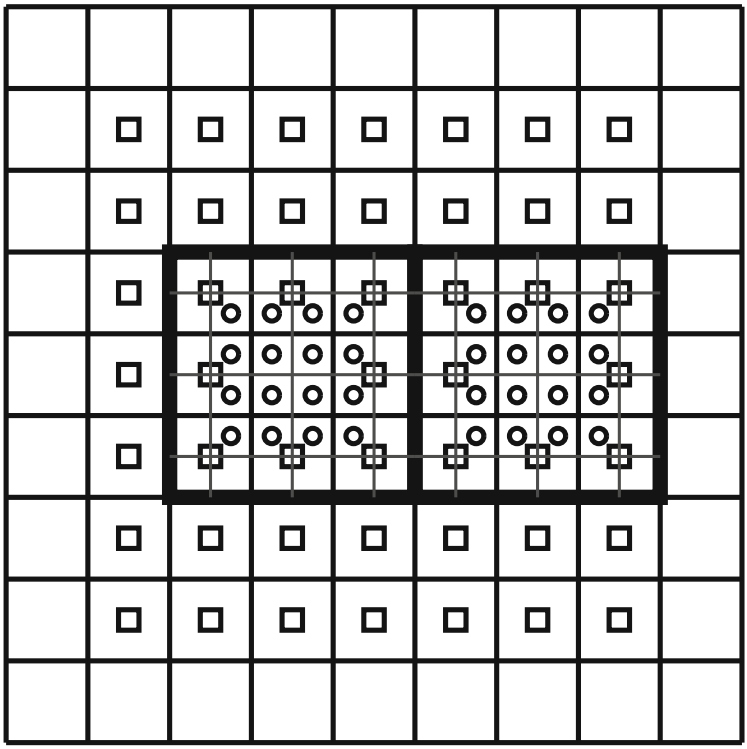

Fig. 4. Kraft was very careful to avoid possible linear dependence issues by specifying that if one old function was refined, those whose support overlapped that of this one should not be refined. They should be left at the original scale. The set of basis functions when two functions as close as permissible are refined is therefore as shown here. This is not the same as would be created by all functions over the new partitioning of the domain.

2.2 Grinspun et al.

In [2] Grinspun et al. used the Loop subdivision scheme to refine a space defined over a triangulation of a curved surface. They report actual analyses of interesting shell structures with non-trivial physics (Fig. 5).

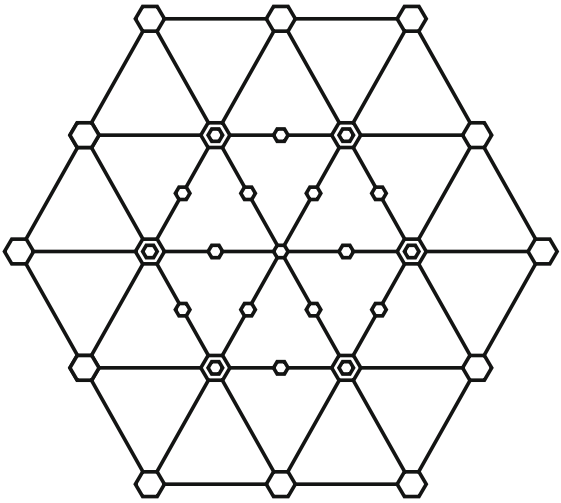

Fig. 5. The method of Grinspun et al. [2]. In this figure (adapted from Figs. 9 and 11 of that paper, with permission) the large hexagons denote centres of basis functions at the coarse level of refinement. They lie on a fairly regular triangular grid. In the centre, a large hexagon is missing but there are small ones denoting the centres of refined basis functions around it. Note that there can be both large and small centred at the same point.

However, their main emphasis, quoted from their abstract, is

...The basic principle of our approach is to refine basis functions, not elements. This removes a number of implementation headaches associated with other approaches and is a general technique independent of domain dimension...

They were able somehow to avoid the linear dependence issue even when supports of refined functions overlapped. If a large region is refined the number of new functions per old one replaced tends down towards four, though for single functions it is three times the valency of the centre of the replaced function plus one.

2.3 Adaptive B-splines

There have been too many papers on this topic to cite them all. These proceedings will contain many papers addressing this topic and references to that literature will be found therein. Two significant contributions are of note, being the Truncated Hierarchical B-splines of Giannelli et al. [3] and the Locally Refined B-splines of Dokken et al. [4]. An excellent comparison of these two with a method like Kraft's but with the functions unscaled by the subdivision ratios, can be found in Johannessen et al. [6].

In the univariate case (illustrated in Fig. 6) both of these recover the partition of unity property which was lost in the original hierarchical work which

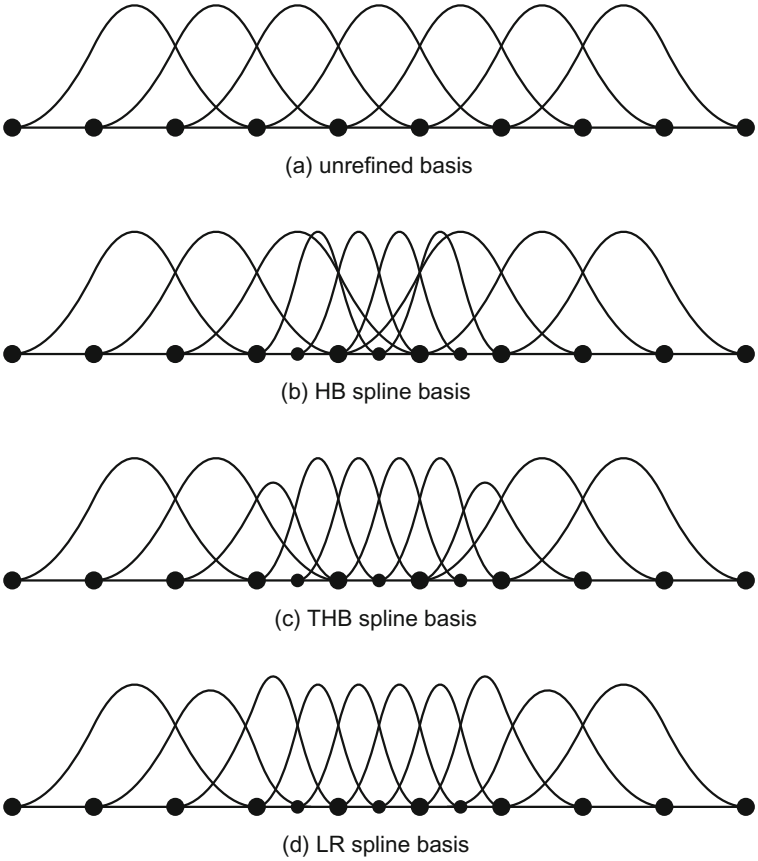

Fig. 6. In the original hierarchical B-spline (HBspline) basis, one function is split into four by knot insertion at three knots (but cf. Fig. 3). The Truncated Hierarchical Bspline (THB-spline) basis [3] modifies two more of the original functions by subtracting some multiple of the refined functions so that their support is reduced back to three spans in the new knot vector, and so that partition of unity is restored. The Locally Refined B-spline (LR-spline) basis [4] takes this a stage further, truncating all functions to the minimum support. In the univariate case this is exactly the same as the non-uniform B-spline basis over the new knot vector.

they cite. In the bivariate case LR-splines still have a problem with linear dependence, although their paper does address how this may be countered. Both sets of authors regard the problem as being finding a basis spanning all piecewise polynomials of a given degree over a given partitioning of the domain.

3 New Contributions

The main contribution here is to reiterate the emphasis of Grinspun et al. More specifically, we

1. Reduce the number of replacement functions down to two per old one. This is detailed in Sect. 3.1.
2. Provide a proof that this process works for all B-spline degrees. This is detailed in Sect. 3.2.
3. Provide a proof that it works for all dimensions of domain. This is detailed in Sect. 3.3.

We consider various dimensions by starting with the univariate case.

3.1 Adaptivity When the Domain Is Univariate

The B-spline degrees are either even or odd. We take these two cases separately

Even Degrees. These are typified by the quadratic. The argument is made in the captions to Figs. 7, 8, 9 and 10, which should be read in sequence as if they were part of the text. The extension to higher even degrees is implied by the descriptions, though not by the illustrations.

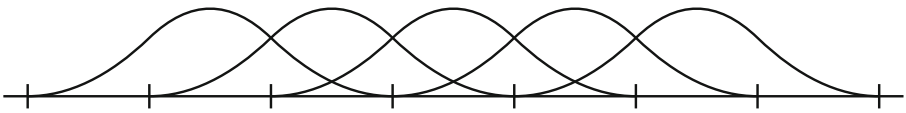

Fig. 7. Suppose that an initial set of quadratic basis functions spanning the solution space is as shown here. This figure shows equal knot intervals, but that is not a pre-condition. After a number of rounds of selective refinement the spacing of the knots will be seriously uneven.

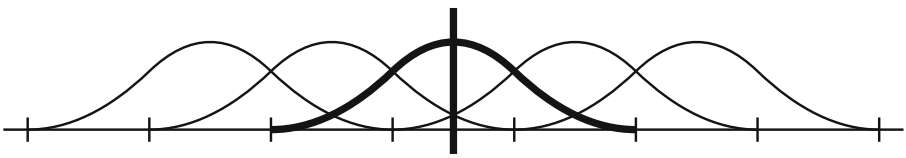

Fig. 8. We can take just one of these functions and insert a single new knot at the midpoint of the central span (all even degree B-splines have an odd number of non-zero spans).

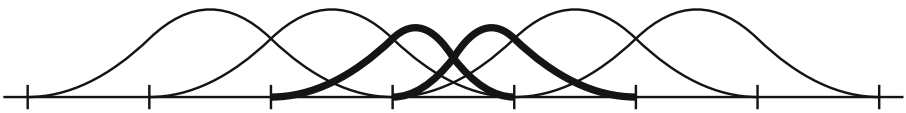

Fig. 9. This knot insertion gives two new functions, which can replace the old. The sum of these two functions is equal to the original, and so the partition of unity property is preserved. In the context where an iterative solver is being used, the coefficients of the new function are both set equal initially to that of the old. This exactly preserves the solution so the refined solution space is a nesting of the old.

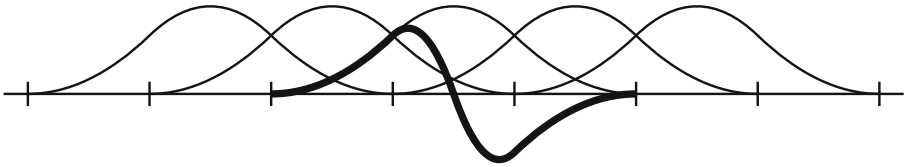

Fig. 10. However, we can 'rotate' the basis within the solution space by rotating the subspace spanned by these two new functions, by taking their sum and difference. This is a non-degenerate transformation which does not alter the rank of the space. The sum of the two new functions is exactly the old one, which has no discontinuity of any derivative at the new knot. Neither do any of the other old functions. The difference function is the only one which has a discontinuity here and it is therefore linearly independent of all the other functions in the extended basis. This is true whether the function being split was part of an initial basis over some coarse partitioning or whether it was the result of many stages of Fig. 9 refinement.

Odd Degrees. These are typified by the cubic. The argument is made in the captions to Figs. 11, 12, 13 and 14, which should be read in sequence. The extension to higher odd degrees is implied by the descriptions there, though not by the illustrations. The situation is slightly more complicated, but the concepts are very similar.

It is a moot point whether the basis of Fig. 9 or that of Fig. 10 (resp., Fig. 13 or Fig. 14) should be used in an actual implementation. The ranks of the Figs. 9 and 10 bases are the same. Each has practical advantages and disadvantages which are not explored further here, except that we note that Figs. 10 and 14

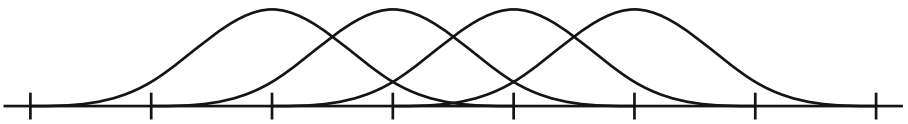

Fig. 11. Suppose that an initial set of cubic basis functions spanning the solution space is as shown here. This figure shows equal knot intervals, but that is not a precondition. After a number of rounds of selective refinement the spacing of the knots will be seriously uneven.

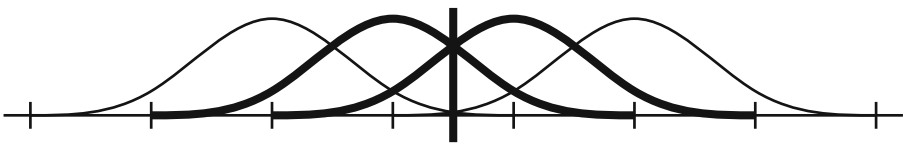

Fig. 12. We can take just two consecutive functions, and insert a single knot at the midpoint of the span between the middle knot of one and the middle knot of the next (all odd degree B-splines have an even number of non-zero spans).

Fig. 13. This knot insertion gives three new functions, which can replace the old two. The sum of the three is equal to the sum of the old two, and so partition of unity is preserved. The detail of the coefficients of the new functions depends on the knot intervals. But the refined solution space is a nesting of the old.

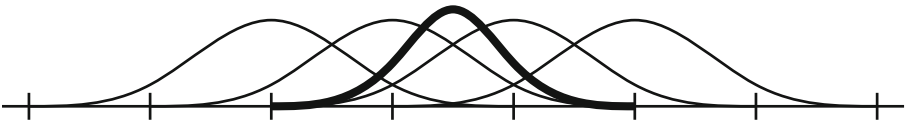

Fig. 14. However, we can again rotate the basis within the solution space by taking appropriate linear combinations of the three new ones, to give the original two, together with one new function which is the only one with a discontinuity at the new knot. It is linearly independent of all other functions in the basis, and the dimension of the space is not altered by the rotation.

no longer preserve partition of unity. This might matter in a modelling context because a coefficient no longer transforms as a point, but as a displacement (see [5]), but in the analysis context it is of no importance whatever. In an iterative solution the coefficient of the new function can be initially set to zero, to match exactly the pre-refinement solution.

The refined space exactly contains the old space as a subspace. This is true whichever of the possible implementations is used.

For even degrees my preference is to use the functions of Fig. 9, though for odd degrees there are more evenly balanced arguments, relating to the necessary infrastructure to steer what refinements can be made.

3.2 Adaptivity When the Domain Is Bivariate

We now assume that all of the original basis functions are tensor products of B-splines in two directions. The argument is made in the captions to Figs. 15, 16, 17, and 18.

The proof of linear independence depends on the fact that a given bivariate function is never split more than once. The splitting introduces a new knot-line and clearly new functions which have different such knot-lines will be linearly independent of each other. If two splits share the same knot-line, but the extents are different in the other direction, then the linear independence of the distribution of discontinuity magnitude in that direction implies linear independence of the new functions. It is useful to notice that, because the split is always of the centre span for even degrees, a given piece of shared knot can only arise at a specific level of the refinement.

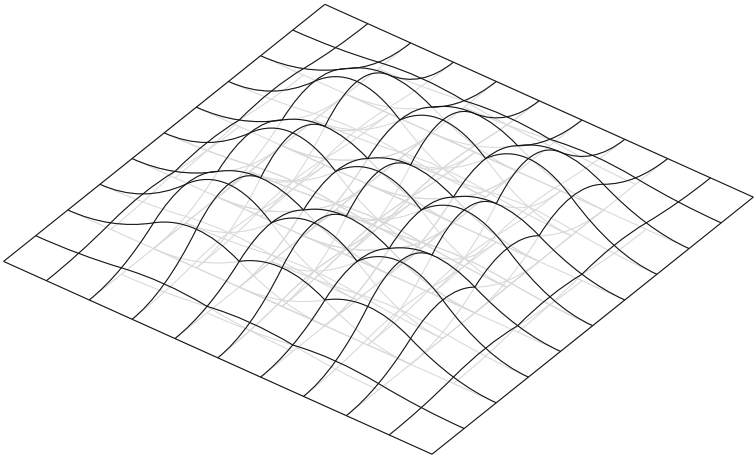

Fig. 15. Suppose that we have an initial set of basis functions over a regular grid. Each basis function is a tensor product of a B-spline in one direction and a B-spline in the other. The two degrees do not have to be equal, although often they will be.

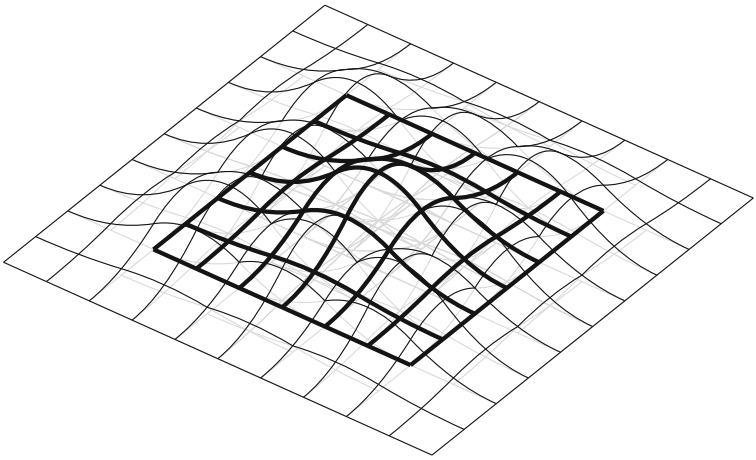

Fig. 16. We can select just one of these basis functions for refinement. In that respect we are following Kraft.

Clearly, in the odd degree case, the two functions split must share the same knots in the other direction. If this is not practical, then the analog of the quadratic case using the span consistently on the same side of the central knot, though less elegant, can be set up.

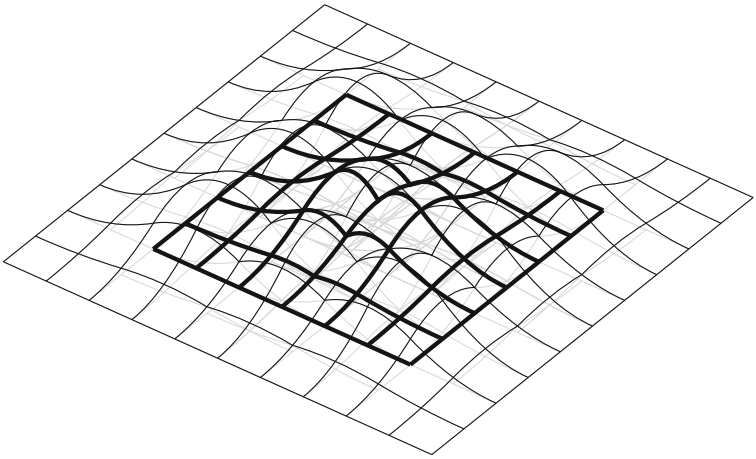

Fig. 17. However, we split in only one direction, not both. We are therefore able to focus the refinement sharply, adding only one dimension to the space, not fifteen. If you want to split in both directions it is necessary to split first in one direction and then both of the new functions in the other. This has the expected symmetry, but because it adds three to the size of the space, takes three refinement operations. The two new functions from a single refinement are given by the tensor product of Fig. 9 (or Fig. 13) with the original B-spline in the other direction.

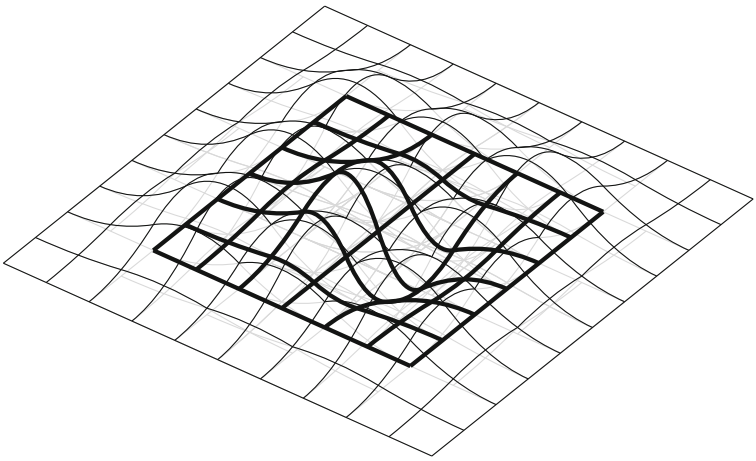

Fig. 18. The rotation in the solution space follows exactly the tensor product of a simple B-spline with the function of Fig. 10.

3.3 Adaptivity When the Domain Has Higher Dimension

Here we take the simple route of specifying that any basis function, which will be a tensor product of as many functions as the dimension of the domain, is split in just one direction, all the other directions keeping their original function. Thus however high the domain dimension becomes, the refinement always increases the size of the solution space by 1.

The proof of linear independence follows that for the bivariate case, except that now the knot insertion is of a piece of knot (hyper)plane. Different hyperplanes cannot be linearly dependent, and different refinements will lead to linearly independent distributions of discontinuity across a common hyperplane.

4 Implementation Issues

An important question is *'How easy is this to incorporate into an existing analysis infrastructure?'*. Even though the scheme is in some ways simpler than the conventional collection of nodes and elements, where the outermost loop during stiffness matrix assembly is through elements, there may be a perceived problem simply because it is different.

4.1 Representation of the Discretisation of the Solution Field

The obvious representation here, assuming isoparametric formulation, is to have a collection of basis function objects, each of which has a number of lists, each as long as 1+degree, one for each independent direction within the domain. These lists contain the domain coordinates of the knots. Thus a triquadratic system over a three-dimensional domain would have a collection of basis functions each represented by three lists of four entries. This has to be supplemented by the map from the parameter domain into the geometric domain, which is used for evaluation of Jacobians. Isogeometric ideas are obviously relevant here, though we are not limited to using the partitioning of the domain which comes from the geometry.

4.2 The Assembly Process

The outermost loops of assembly are now most conveniently taken through pairs of basis functions. Each possible pair corresponds to a single entry in the stiffness matrix, and so using this structure lends itself to effective parallelisation on multi-processor machines with little shared memory.

For each pair we have to do an integration over the intersection of the supports. Here there is a nice surprise: although the domain may appear to be totally splintered by different refinements, the number of polynomial pieces over which the integration must be done is bounded by the number of knots active in the two basis functions forming the pair. They don't see knots which are only discontinuities in other basis functions.

4.3 Solution

There is no reason why, once the stiffness matrix and right hand sides are formed, the solution process has to be any different. However, because of the exact nesting of the spaces, there will be a lot of coherence between successive solutions. This is clearly evident in the case of iterative solution.

However, it is also possible to exploit this coherence in the case of explicit solution by factorisation. When the total number of refinements is small relative to the total number of freedoms, it is possible merely to update the factors at a cost significantly less than starting again.

4.4 Error Estimation

Here for maximum efficiency, we need information not only about the magnitude of the local error, but also about which direction the refinement should be made in. The obvious rule of thumb is to split in the direction in which the derivative of error is largest. (Think about the S-shaped function in Fig. 10. Adding it to the solution space will obviously be good for correcting the derivative of the error.)

4.5 Result Extraction

Because there are no longer any elements which can act as a scaffolding for display of results, this process becomes more complex. At any point where the result has to be evaluated you need to know which basis functions include that point in their support. Some kind of spatial indexing is likely to be indispensable here. This can also give some benefit during the assembly, when identifying which pairs of basis functions have a non-zero overlap could become a significant task.

5 Discussion

'Is this just the same as any of the hierarchical formulations which have been considered, over the last decade or so, which start with a partitioning of the domain and then work out a linearly independent basis for the space of all piecewise polynomials of given continuity over that partitioning?'

No, it is not.

It starts instead with the process of refinement of the basis and a partitioning is the result. That partitioning as such plays a minor role in the solution of any specific problem. The size of the space which the basis spans may be significantly smaller than the maximal linear independent space, as we have seen in Fig. 4, where the maximal dimension over the refined partitioning is six more than Kraft's basis.

'Are you not concerned that you are not using the full space of functions over the partitioning? What does this do to the approximation power?'

Theoretical asymptotic approximation order is not the critical issue in practical use. What matters is whether the solution is sufficiently accurate (usually errors of the order of 10^{-2} in stresses), and whether it is achieved at minimal cost.

Selecting basis functions among those possible is a strength, not a weakness, because the adding of new functions to the basis is done as a response to where the error needs to be reduced. Every extra freedom increases the cost of solution: if there is evidence that it will improve the solution then it can be added: if there is not, you don't want it. Figure 19 illustrates this.

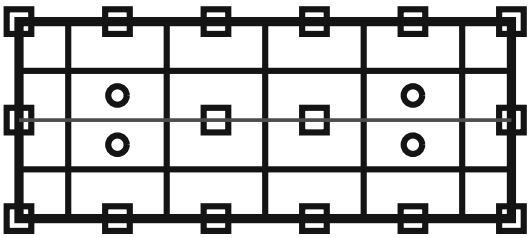

Fig. 19. An example where the full space is wasteful. In this figure just two basis functions in an original 3×6 grid have been refined. The full space over the resulting partitioning of the domain would have had to split all of the functions in the original middle row, making the total number of freedoms four larger. If the error estimator had requested only the two functions to be refined, the extras would have contributed to cost without making the solution any better. This figure may also be compared with Fig. 4.

'This looks OK for the first addition of a new function, but what happens when nearby basis functions have already been split?'

This is very straightforward for even degrees. Think in terms of Fig. 9, where all basis functions are indeed B-splines over their own knot vectors. Any such B-spline can be split independently of anything else around it. Figure 20 shows the univariate case, but there is no extra complication for higher even degrees.

Odd degrees appear more complicated, and indeed they are. However, the new functions in Fig. 14 are still B-splines over their own knot vectors. The central knot is new and its position and extent are not shared by any other functions. This is all that matters. The choice of the other knots is actually arbitrary. In principle we could add a basis function designed to match the local error behaviour: the new knot does not even need to be aligned with the existing mesh directions. That is an exciting prospect far too speculative for this paper. The message here is that almost any choices made in designing software will work. The problem is that there are too many possible effective answers, not the failure of any of them.

'Something that is usually unwanted when dealing with adaptivity is a not bounded number of non-zero basis functions on a certain point of the domain.

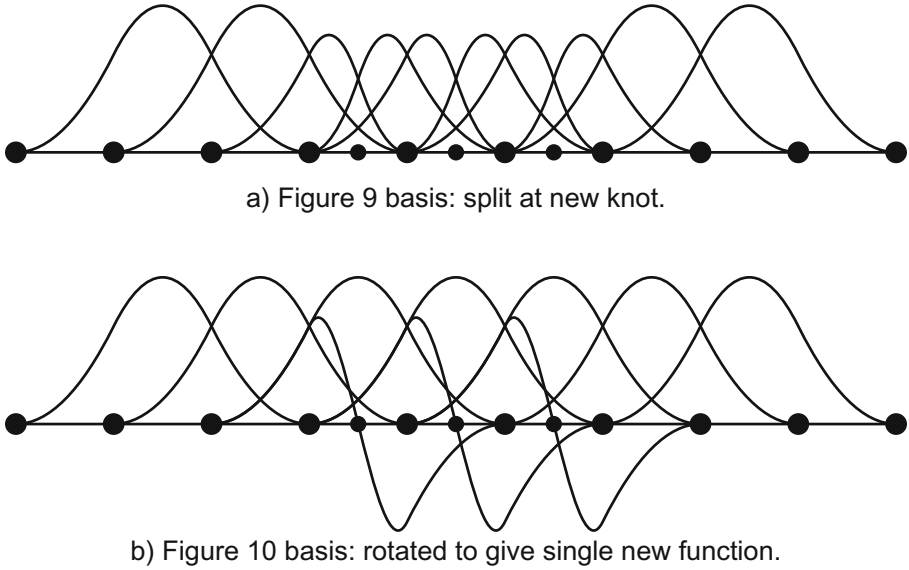

Fig. 20. It is interesting to compare the proposed quadratic functions with those shown in Fig. 6. The upper figure corresponds to Fig. 9 and the lower to Fig. 10.

If one is keeping refining in the same area it seems that the proposed solution does not overcome this problem (while this is partially addressed by the reduced support of THB-spline, or solved by the minimal support of LR B-splines).'

Each entry in the stiffness matrix, representing the interaction of two freedoms, has a bounded number of polynomial regions for integration.

The actual depth of refinement depends on the actual errors found by the error estimator. This should only grow significantly if the solution field contains components of significantly high spatial frequency which, for elliptic problems will happen only near the boundary.

If functions share knots in the other direction(s) rotation of the basis can bring it back to a narrower bandwidth.

'In order to have good numerical properties (e.g., condition numbers of related matrices) certain mesh grading is usually preferable. In the multivariate setting, if the knot insertion is always performed in the same direction, elongated elements (or equivalently, elongated B-spline elements if one prefers to not refer to domain partitioning) may be created. This is why it is common practice to perform dyadic refinement on the marked elements (even if also with the other hierarchical construction the refinement can be carried over along only one direction).'

As is the case with the other approaches, when bad aspect ratios are about to be produced, additional refinement in the other direction(s) can be invoked. The question is the monitoring of the aspect ratio, not the availability of a solution.

'Section 4 presents some short comments on critical aspects of an adaptive framework without entering into the details of any of them. From a practical point of view the effectivity of this construction that adds only one basis function at the time, should be verified. There are many open questions concerning the resulting adaptive scheme (function based error estimation which takes into account derivative information, some refinement strategies) which may effect the performance of the method.'

Yes, of course, there are a lot more issues to adaptivity than just the basis. However, these are not significantly affected by the message of this paper. I didn't want to dilute that message by details of loosely-coupled aspects. What is worth stressing is that although the presentation here talks about 'one at a time', there is no reason to invoke a complete re-solution after each one. If the error estimator finds a lot of places where additional freedoms are needed, then of course all those functions need to be added.

6 Conclusions

An approach to adaptive refinement has been described with a number of significant advantages over previous methods. It follows the ideas of Kraft and of Grinspun et al., of starting from the basis rather than the partitioning of the domain, but extends these by making the refinement much more selective. Each refinement adds only one freedom to the system to be solved, and (although it is certainly possible to make several refinements before attempting another solution) this keeps the cost of solution as low as possible. Furthermore, because the refined space is exactly nested (as is the case for the other approaches mentioned above), the next solution can be regarded as an update which may be a lot cheaper than a re-solution.

References

1. Kraft, R.: Adaptive and linearly independent multilevel B-splines. In: Le Méhauté, A., Rabut, C., Schumaker, L. (eds.) Surface Fitting and Multiresolution Methods, pp. 209–218. Vanderbilt University Press (1997)
2. Grinspun, E., Krysl, P., Schröder, P.: A simple framework for adaptive simulation. In: Proceedings of SIGGRAPH 2002 (ACM Transactions on Graphics), pp. 281–290 (2002)
3. Giannelli, C., Jüttler, B., Speleers, H.: THB-splines: the truncated basis for hierarchical splines. CAGD **29**(7), 485–496 (2012)
4. Dokken, T., Lyche, T., Pettersen, K.F.: Polynomial splines over locally refined box-partitions. CAGD **30**(3), 331–356 (2013)
5. Kosinka, J., Sabin, M.A., Dodgson, N.A.: Control vectors for splines. CAD **58**, 173–178 (2015)
6. Johannessen, K.A., Remonato, F., Kvamsdal, T.: On the similarities and differences between classical hierarchical, truncated hierarchical and LR B-splines. CMAME **291**, 64–101 (2015)