# 4 Random Fuzzy Numbers and Vectors

## 4.1 Introduction

We first discuss how we plan to produce random triangular (trapezoidal) fuzzy numbers. Then we define quadratic fuzzy numbers and show how we can produce random sequences of these fuzzy numbers using sequences of random vectors of real numbers. Random quadratic fuzzy numbers are used in Chapters 10 and 27. In the last section we consider producing random sequences of fuzzy vectors $\overline{V} = (\overline{X}_1, ..., \overline{X}_n)$ where the $\overline{X}_i$ are triangular/trapezoidal fuzzy numbers or quadratic fuzzy numbers. These random sequences of fuzzy vectors have applications throughout these chapters; triangular fuzzy numbers (TFNs) in Chapters 11-12, 15-17; trapezoidal fuzzy numbers (TrFNs) in Chapter 13; and QBGFNs (defined below) in Chapters 6-10, 19 and 27. We also abbreviate "fuzzy numbers" as FNs. Trapezoidal shaped fuzzy numbers have applications in Chapters 13, 20-22 and 26.

## 4.2 Random Triangular/Trapezoidal Fuzzy Numbers

First consider producing a random sequence of triangular fuzzy numbers in an interval $[a, b]$. We want to create the sequence $\overline{X}_i = (x_{i1}/x_{i2}/x_{i3})$, $i = 1, 2, 3, ...,$ with $\overline{X}_i \in [a, b]$ all $i$. Using our Sobol quasi-random number generator (Chapter 3), we take three consecutively generated numbers and order them. Thus we make a sequence of random vectors $v_i = (z_{i1}, z_{i2}, z_{i3})$, $z_{i1} < z_{i2} < z_{i3}$, $i = 1, 2, 3, ...$ in $[0, 1]^3$. Then set $x_{ij} = (b - a)z_{ij} + a$, $j = 1, 2, 3$ and $i = 1, 2, 3, ....$

Now consider making a random sequence of trapezoidal fuzzy numbers in $[a, b]$. Assume the sequence is $\overline{X}_i = (x_{i1}/x_{i2}, x_{i3}/x_{i4})$, $i = 1, 2, 3, ....$ Using our Sobol quasi-random number generator (Chapter 3) we take four consecutively generated numbers and order them. Thus we make a sequence of random vectors $v_i = (z_{i1}, z_{i2}, z_{i3}, z_{i4})$, $z_{i1} < z_{i2} < z_{i3} < z_{i4}$, $i = 1, 2, 3, ...$ in $[0, 1]^4$. Then set $x_{ij} = (b - a)z_{ij} + a$, $j = 1, 2, 3, 4$ and $i = 1, 2, 3, ....$ We will use these random FNs in Section 4.5.

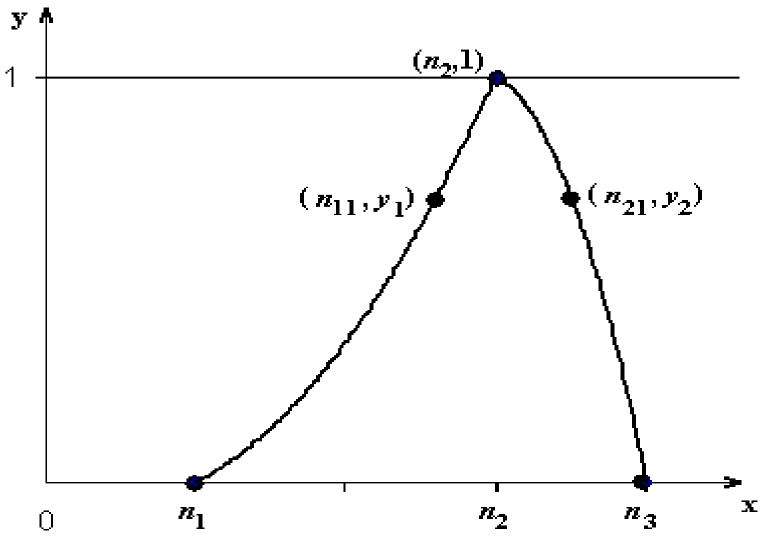

**Fig. 4.1.** Random Quadratic Fuzzy Number $\overline{N}$: Case 1

## 4.3 Random Quadratic Fuzzy Numbers

### 4.3.1 Generated from Implicit Quadratic Functions

Let $\overline{N} \approx (n_1/n_2/n_3)$, or $\approx (n_1/n_2, n_3/n_4)$, in $[0,1]$ be a triangular/trapezoidal shaped fuzzy number. In this section we will discuss how we plan to produce a sequence of random triangular/trapezoidal shaped fuzzy numbers in some interval $[a,b]$. We first make $\overline{N}$ in $[0,1]$ and multiply by $b-a$ and add $a$ to get it into $[a,b]$. We always start out trying to make a triangular shaped fuzzy number but sometimes, as we show below, it turns out to be a trapezoidal shaped fuzzy number.

Let $y = f_1(x)$ denote the function that makes the left side of the membership function $y = \overline{N}(x)$, $0 \leq y \leq 1$, $n_1 \leq x \leq n_2$. We assume that $f_1(x)$ is continuous and strictly increasing with $f_1(n_1) = 0$ and $f_1(n_2) = 1$. Next let $y = f_2(x)$ denote the function that makes the right side of the membership function $y = \overline{N}(x)$, $0 \leq y \leq 1$, $n_2 \leq x \leq n_3$. We assume that $f_2(x)$ is continuous and strictly decreasing with $f_2(n_2) = 1$ and $f_2(n_3) = 0$. Notice that if we substitute $\alpha$ for $y$ an $\alpha$-cut of $\overline{N}$ can be written $[f_1^{-1}(\alpha), f_2^{-1}(\alpha)]$.

In this chapter we will use quadratic functions for the $f_i(x)$. Let $a_{i1}x^2 + a_{i2}x + a_{i3} = f_i(x)$, $i = 1, 2$. We may extend the results to higher order polynomials. Now choose $n_{11}$ and $n_{21}$ so that $n_1 < n_{11} < n_2$, $n_2 < n_{21} < n_3$, and then choose $y_1$, $y_2$ in $(0,1)$. The left side of $\overline{N}$ will be determined by the three points $(n_1, 0)$, $(n_{11}, y_1)$, $(n_2, 1)$ because these three points, assuming they do not lie in a straight line, uniquely determine the $a_{1j}$ in $y = a_{11}x^2 + a_{12}x + a_{13}$. The right side of $\overline{N}$ will be determined by the three points $(n_2, 1)$, $(n_{21}, y_2)$, $(n_3, 0)$ because these three points, assuming they do not lie in a straight line, uniquely determine the

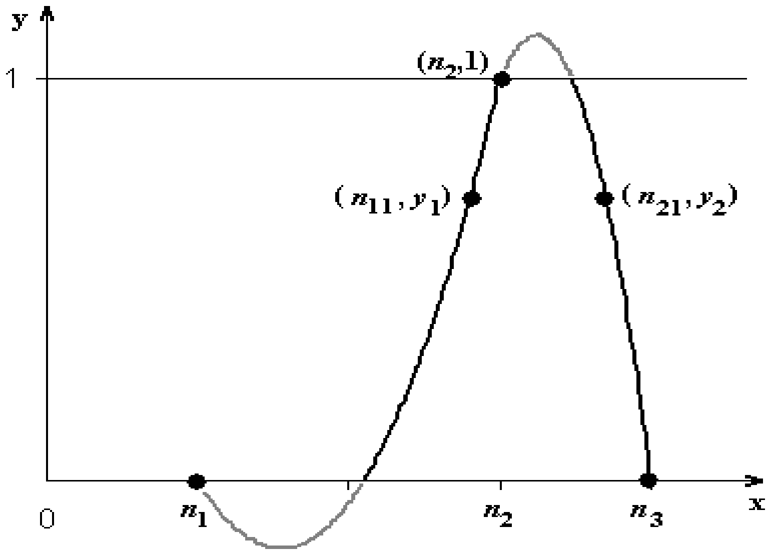

**Fig. 4.2.** Random Quadratic Fuzzy Number $\overline{N}$: Case 2

$a_{2j}$ in $y = a_{21}x^2 + a_{22}x + a_{23}$. So we require the seven numbers $n_1$, $n_{11}$, $y_1$, $n_3$, $n_{21}$, $y_2$ and $n_3$ to construct our triangular shaped fuzzy number $\overline{N}$. We will call these fuzzy numbers quadratic fuzzy numbers because they have quadratic membership functions. In the case of Figure 4.1 it is also a triangular shaped fuzzy number. As shown in Figure 4.2 this method may result in a trapezoidal shaped fuzzy number or a triangular shaped fuzzy number. One can identify nine different cases of quadratic fuzzy numbers; we have shown one case which results in a trapezoidal shaped fuzzy number for which the left support and the right extent of the core must yet be determined.

To randomly generate a Case 1 $\overline{N}$ in $[0,1]$ we randomly produce random numbers $x_1, ..., x_7$ in $[0,1]$, using our Sobol quasi-random number generator (Chapter 3), giving the random vector $w = (x_1, ..., x_7) \in [0,1]^7$. In $w$ first randomly choose two values say, for example, $x_3$ and $x_6$. Then set $y_1 = x_3$ and $y_2 = x_6$. Now order the remaining five numbers from smallest to largest giving, for example, $x_5 < x_2 < x_7 < x_1 < x_4$. Then define $n_1 = x_5$, $n_{11} = x_2$, $n_2 = x_7$, $n_{21} = x_1$ and $n_3 = x_4$. We now have the five points to get triangular shaped fuzzy number $\overline{N}$. See Figure 4.1.

However, there are problems with constructing quadratic fuzzy numbers this way. The graph of the left side of the quadratic fuzzy number between $n_1$ and $n_2$ in Figure 4.1 may go above $y = 1$ or below $y = 0$, but not both. This may happen as in Figure 4.2. Then we use either $min\{f_1(x), 1\}$ or $max\{f_1(x), 0\}$ for the left side. Also, the graph of the right side of the quadratic fuzzy number between $n_2$ and $n_3$ in Figure 4.1 may go above $y = 1$ or below $y = 0$, but not both. This may happen as in Figure 4.2. Then we use either $min\{f_2(x), 1\}$ or $max\{f_2(x), 0\}$ for the right side. This produces eight more cases of quadratic fuzzy numbers, some trapezoidal shaped fuzzy numbers having quadratic sides, some triangular shaped fuzzy numbers with quadratic sides. But in all cases the sides of the fuzzy number are described by quadratic functions. These cases make up all our quadratic fuzzy numbers to be randomly generated from random vectors $w = (x_1, ..., x_7)$.

Because of the problems discussed above, curves going above (below) the horizontal line $y = 1$ ($y = 0$), we will not use this method of generating random quadratic fuzzy numbers in this book. Instead, we will employ the procedure outlined in the next section.

### 4.3.2 Generated from Parametric Quadratic Functions, Bézier Fuzzy Numbers

We now show that we can also generate random triangular shaped fuzzy numbers using random vectors $w = (x_1, ..., x_5) \in [0,1]^5$ of length five. The definition and properties of Bézier generated fuzzy numbers (BGFNs) is a result of research on random fuzzy numbers done by Leonard Jowers at the University of Alabama at Birmingham [2]. Bézier generated fuzzy numbers have a 100% yield of triangular shaped FNs. What happened in Figure 4.2 will not occur now. Methods for two types of BGFNs are given in [3]. Here we only discuss those which result in FNs which have quadratic membership functions, quadratic Bézier generated fuzzy numbers (QBGFNs) are FNs whose membership functions, left and right, are defined by parabolas generated from Bézier curves [1]. Yet another representation having membership functions consisting of hyperbolas, ellipses, a line and a parabola may be generated from weighted quadratic rational Bézier curves; however, those conic Bézier generated fuzzy numbers (CBGFNs) do not generate quadratic membership functions in their explicit forms and are not used in this book.

A full explanation of QBGFNs is lengthy and beyond the scope here. However, a full explanation is available in [3], which includes how one may generate QBGFNs (parabolic) or CBGFNs, and recover their explicit forms.

Creation of Bézier curves can be visualized through de Castlejau's Algorithm [1]. Given three control points, $b_0, b_1, b_2$, we create a curve as a moving point, $b_0^2$ (Figure 4.3), on a moving line, $\overline{b_0^1 b_1^1}$. Both move, maintaining the relationship $\overline{b_0 b_0^1}$ is to $\overline{b_0^1 b_1}$, as $\overline{b_1 b_1^1}$ is to $\overline{b_1^1 b_2}$, as $\overline{b_0^1 b_0^2}$ is to $\overline{b_0^2 b_1^1}$. The point $b_0^2$ is determined by repeated linear interpolation.

A quadratic Bézier curve is an arc length parameterization, $t$ restricted to $[0,1]$, of a parabola. Where $b_i$ is $\begin{bmatrix} x_i \ y_i \end{bmatrix}^T$, we have

$$\begin{bmatrix} x(t) \\ y(t) \end{bmatrix} = (1-t)^2 b_0 + 2(1-t)t b_1 + t^2 b_2. \tag{4.1}$$

$b_0, b_1$, and $b_2$ are coefficients to Bernstein polynomials, $B_{k,n}(t) = \binom{n}{k} t^k (1-t)^{n-k}$, of degree 2; that is, $B_{0,2} = (1-t)^2$, $B_{1,2} = 2(1-t)t$, and $B_{2,2} = t^2$. Bernstein polynomials of degree $n$ are the terms of the expansion of $[(1-t)+t]^n$.

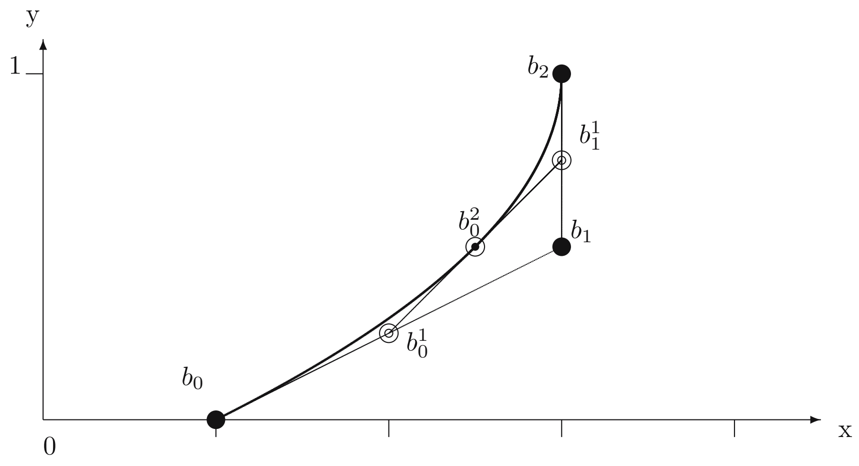

**Fig. 4.3.** A point, $b_0^2$, in de Castlejau's Algorithm

We now define quadratic Bézier generated fuzzy numbers (QBGFNs) as fuzzy numbers (FNs) whose membership function is a pair of quadratic Bézier curves [2]. This representation requires a vector of length 5. We produce a quadratic fuzzy number from a "random" vector $v = (z_1, ..., z_5)$ generated from some 'random' method. Three elements of a vector of length 5 define a TFN template of a QBGFN. The two additional elements define the shape of the left and right membership functions.

First we specify the range over which we allow the support, which we now assume to be $[0, M]$, $M > 0$. A stream of crisp random integers gives a vector $v$ of length 5 and we first map each component into [0,1]. So assume $v = (z_1, ..., z_5)$, $z_i \in [0, 1]$ all $i$. Let the final "random" vector, used for the quadratic fuzzy number $\overline{A}$ be $w = (x_1, ..., x_5)$. However, our construction is to first determine the support and vertex of a TFN, then a parameter for left (right) side of the membership function. The first three elements of the vector $v$ are sorted to create $z_2 < z_1 < z_3$, then $x_1 = Mz_2$, $x_2 = Mz_1$, $x_3 = Mz_3$ and the TFN is $(x_1/x_2/x_3)$ in $[0, M]$. Next $z_4$ ($z_5$) are transformed into $x_4$ ($x_5$) which produce the quadratic membership function for the left (right) side the quadratic fuzzy number $\overline{A}$.

The left membership function is defined in the following manner. $z_4$ is mapped to $x_4$ by $x_4 = (2z_4 - 1) \times ((x_2 - x_1) + 1)$ (similarly, $x_5$ is computed from $z_5$ as $x_5 = (2z_5 - 1) \times ((x_3 - x_2) + 1))$. $x_4$ will be in $[-(x_2 - x_1 + 1), (x_2 - x_1 + 1)]$. If $x_4 < 0$, $x_4$ defines a Bézier control point on the path counterclockwise from $(x_1, 0)$ to $(x_2, 0)$ to $(x_2, 1)$. $|x_4|$ is the Manhattan distance from $(x_1, 0)$ along that path. See $b_0$, $b_1$ and $b_2$ in Figure 4.4, the Bézier control points for the left membership function. One may see that, if $x_4 < 0$, the Bézier control points of the left membership are either $\{(x_1, 0), (|x_4|, 0), (x_2, 1)\}$ for $|x_4|$ the distance between $(x_1, 0)$ and $(x_2, 0)$, or $\{(x_1, 0), (x_2, |x_4 - (x_2 - x_1)|), (x_2, 1)\}$ for $|x_4|$ greater than the distance between $(x_1, 0)$ and $(x_2, 0)$, Otherwise, $x_4 \geq 0$ similarly defines a Bézier control point on the path clockwise from $(x_1, 0)$ to $(x_1, 1)$ to $(x_2, 1)$.

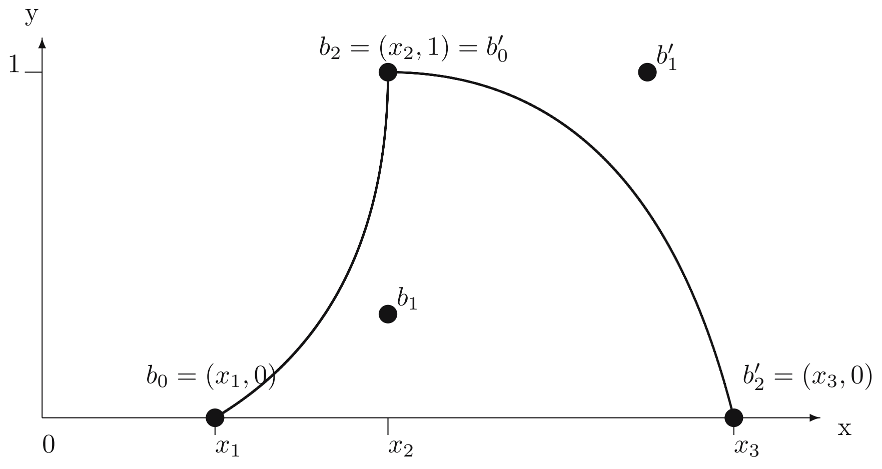

**Fig. 4.4.** A Parabolic QBGFN Fuzzy Number $\overline{A}$

The same process is used to determine the Bézier control points $b'_0$, $b'_1$ and $b'_2$ (see Figure 4.4) for the right membership function.

In [3] we show how a Bézier curve may be converted into its implicit form; i.e., $y = f(x)$. Additionally there, we show a quadratic Bézier is converted to an implicit quadratic polynomial function.

These QBGFNs are used extensively in Chapters 6-10 and their use is proposed in Chapters 19 and 27.

## 4.4 Comparison of Random Fuzzy Vectors

In our evaluation of streams of FNs, to be continued in the next chapter, we investigated our algorithms for comparing fuzzy numbers. For Buckley's method (Section 2.6.1) we evaluated the effect of choices of threshold values. Buckley thresholds of 0.7, 0.8, and 0.9 were evaluated. 0.7 generated an excessive number of "equal" results. 0.9 appeared too much like a vertex comparison. We determined that a Buckley threshold of 0.8 provided an acceptable comparison.

We also investigated differences among our three chosen comparison methods, Buckley, Kerre (Section 2.6.2), and Chen (Section 2.6.3). $10,000$ FNs created using various crisp input streams were evaluated. In Table 4.1 "Type FN" refers to the type of input stream; for example, "Sobol 10" is for QBGFNs generated from Sobol quasi-random numbers which had been generated 10 at a time (as 10-tuples). Given two fuzzy numbers $\overline{M}$ and $\overline{N}$, using any one of the three comparison methods, only one of the ordering results $\overline{M} < \overline{N}$, or $\overline{M} \approx \overline{N}$, or $\overline{M} > \overline{N}$ can be true. "All Agree" are the number times all three methods agreed upon the order of a pair of FNs. "Non-agree" represents the rest of the $10,000$. "B&K agree" are the number of times that Buckley and Kerre agree when Chen did not. "B&C agree" are the number of times that Buckley and Chen agree when Kerre did not. "K&C agree" are the number of times that Kerre and

**Table 4.1.** Comparison of Buckley (0.8), Kerre, and Chen Methods

| Type FN | All agree | Non-agree | B&K agree | B&C agree | K&C agree | All disagree |
|---|---|---|---|---|---|---|
| QBGFN | | | | | | |
| Pseudo | 7984 | 2016 | 289 | 267 | 1424 | 36 |
| True | 8032 | 1968 | 249 | 309 | 1384 | 26 |
| Faure 5 | 6402 | 3598 | 105 | 100 | 3363 | 30 |
| Halton 5 | 8151 | 1849 | 240 | 275 | 1284 | 50 |
| Niederreiter 5 | 8471 | 1529 | 180 | 176 | 1157 | 16 |
| Sobol 5 | 8667 | 1333 | 86 | 121 | 1109 | 17 |
| Faure 10 | 8019 | 1981 | 280 | 257 | 1410 | 34 |
| Halton 10 | 8039 | 1961 | 275 | 259 | 1390 | 37 |
| Niederreiter 10 | 7953 | 2047 | 286 | 267 | 1472 | 22 |
| Sobol 10 | 7956 | 2044 | 283 | 288 | 1440 | 33 |
| Faure 15 | 8155 | 1845 | 200 | 264 | 1356 | 25 |
| Halton 15 | 7977 | 2023 | 307 | 270 | 1417 | 29 |
| Niederreiter 15 | 7961 | 2039 | 300 | 270 | 1425 | 44 |
| Sobol 15 | 8022 | 1978 | 276 | 267 | 1398 | 37 |
| TFN | | | | | | |
| Pseudo | 8499 | 1501 | 203 | 202 | 1050 | 46 |
| True | 8461 | 1539 | 211 | 236 | 1065 | 27 |

Chen agree when Buckley did not. "All disagree" are the number of times that each gave a different result. An example of an "All disagree" situation could be: Buckley's method says $\overline{M} < \overline{N}$, Kerre's procedure implies $\overline{M} \approx \overline{N}$, but Chen's way gives $\overline{M} > \overline{N}$.

Table 4.1 shows that results did not greatly vary among FNs generated from our various streams of crisp random numbers. We also found that results did not vary greatly between QBGFNs and TFNs. We found that Kerre's method (Section 2.6.2) and Chen's method (Section 2.6.3) agreed on average for almost 95% of the comparisons. In Table 4.1, one may add columns "All agree" and "K&C agree", for any row, to see that for the $10,000$ comparisons about $9,500$ show the same result for Kerre's and Chen's methods. Now both Kerre's method and Chen's method are used in Chapters 6-8, so we will not be too surprised if they give similar results. Buckley's method is not used in Chapters 6-8 but will be employed in the rest of the book.

## 4.5 Random Fuzzy Vectors

We first look at obtaining sequences of random fuzzy vectors whose components are all quadratic fuzzy numbers. Suppose we want $\overline{V}_i = (\overline{X}_{i1}, \overline{X}_{i2}, \overline{X}_{i3})$, $i = 1, 2, 3, ....$ Using our quasi-random number generator we get a sequence of random vectors $v_i = (x_{i1}, ..., x_{i5})$, $i = 1, 2, 3, ....$ We take $v_1$ for $\overline{X}_{11}$, $v_2$ for $\overline{X}_{12}$, $v_3$ for $\overline{X}_{13}$, etc. We could also make $v_i$ have length 15 and take the first five components for $\overline{X}_{i1}$, the second five components for $\overline{X}_{i2}$, etc. We will usually use the first method.

If we wanted a sequence of random fuzzy vectors $\overline{V}_i$ whose components are all TFNs we may use random crisp vectors $v_i$ of length nine. Take the first three components to make $\overline{X}_{i1}$, etc. For random fuzzy vectors of trapezoidal fuzzy numbers we can use random crisp vectors of length 12 using the first four for $\overline{X}_{i1}$, etc.

## References

1. Farin, G.: Curves and Surfaces for CADG, 5th edn., pp. 219–224. Academic Press, London (2002)
2. Jowers, L.J.: Fuzzy Numbers, A Novel Representation Using Bézier Curves. In: UAB Graduate Student Research Day 2005, Birmingham, AL (March 4, 2005)
3. Jowers, L.J., Buckley, J.J., Reilly, K.D.: Representation of Conic and Parabolic Fuzzy Numbers Using Bézier Curves. In: Proceedings IPMU 2006. 11th International Conference of Information Processing and Management of Uncertainty in Knowledge Based Systems, Paris, France, July 2, 2006, p. 6 (2006)