University of Ostrava

Institute for Research and Applications of Fuzzy Modeling

Computational Complexity of Discrete Fuzzy Transform

Viktor Pavliska

Research report No. 113

2006

Submitted/to appear:

Internal publication

Supported by:

MSM 6198898701 and project 1M0572 of the MŠMT ČR

University of Ostrava
Institute for Research and Applications of Fuzzy Modeling
30. dubna 22, 701 03 Ostrava 1, Czech Republic

tel.: +420-59-6160234 fax: +420-59-6120 478
e-mail: viktor.pavliska@osu.cz

Abstract

The aim of this paper is to show estimations of computational time complexity of fuzzy approximation method called fuzzy transforms for approximation of continuous function. This technique was introduced by prof. Perfilieva, I. in [1] or [2]. All estimations of computational complexity are based on the implementation of mentioned algorithm used in [3] and documented in technical report [4].

Key words: fuzzy transform, algorithmic complexity.

1 Introduction

The main idea of fuzzy transform approximation technique consists in the replacement of an continuous function on a real closed d-dimensional interval by its discrete representation (using the direct F-transform). Afterwards, the discrete representation is transformed back to the space of continuous functions (using the inverse F-transform). The result, obtained by applying both F-transforms is a good simplified approximation of an original function.

2 Definitions

All following definitions were taken from [5].

2.1 Fuzzy Partition of the Universe

The core idea of the fuzzy transform technique is a fuzzy partition of the universe. Let a fuzzy partition of $[a_j, b_j]$ be given by basic functions $\mathbf{A}_1^j, \ldots, \mathbf{A}_{n_j}^j \subsetneq [a_j, b_j], n > 2$ for $j = 1, \ldots, r$. Then the *fuzzy partition of* $\mathcal{D}^r$ is given by the fuzzy Cartesian product $\{\mathbf{A}_1^1, \ldots, \mathbf{A}_{n_1}^1\} \times_\odot \{\mathbf{A}_1^2, \ldots, \mathbf{A}_{n_2}^2\} \times_\odot \cdots \times_\odot \{\mathbf{A}_1^r, \ldots, \mathbf{A}_{n_r}^r\}$ w.r.t. the product t-norm of these r fuzzy partitions. If all fuzzy partitions of particular axes are uniform (with a symmetry) then also the overall fuzzy partition is uniform (with a symmetry).

2.2 Discrete F-Transform

Suppose that an original function $f(x_1, \ldots, x_r)$ is known (may be computed) only at some nodes. Let $\{\mathbf{A}_1^1, \ldots, \mathbf{A}_{n_1}^1\} \times_\odot \{\mathbf{A}_1^2, \ldots, \mathbf{A}_{n_2}^2\} \times_\odot \cdots \times_\odot \{\mathbf{A}_1^r, \ldots, \mathbf{A}_{n_r}^r\}$ be a fuzzy partition of $\mathcal{D}^r$ and let a function $f : \mathcal{D}^r \to \mathbb{R}$ be known at nodes $(p_1^1, \ldots, p_1^r), \ldots, (p_N^1 \ldots, p_N^r)$ such that for each $(i_1, \cdots, i_r)$ where $i_j = 1, \ldots, n_j$ and $j = 1, \ldots, r$, there exists $k = 1, \ldots, N$: $\mathbf{A}_{i_1}^1(p_k^1) \cdots \mathbf{A}_{i_r}^r(p_k^r) > 0$. We say that the ν-tuple $[F_{i_1 \cdots i_r}]$ of real numbers is *the discrete direct F-transform* of f w.r.t. the given fuzzy partition if

$$F_{i_1 \cdots i_r} = \frac{\sum_{k=1}^N f(p_k^1, \ldots, p_k^r) \mathbf{A}_{i_1}^1(p_k^1) \cdots \mathbf{A}_{i_r}^r(p_{i_r}^r)}{\sum_{k=1}^N \mathbf{A}_{i_1}^1(p_k^1) \cdots \mathbf{A}_{i_r}^r(p_{i_r}^r)} \tag{1}$$

for each r-tuple $i_1 \cdots i_r$, where $\nu = n_1 \cdot n_2 \cdots n_r$ is count of all fuzzy transform components.

2.3 Inverse Discrete F-Transform

Let $F^{(r)}[f]$ be the direct F-transform of $f \in \mathcal{C}(\mathcal{D}^r)$ w.r.t. a given fuzzy partition $\{\mathbf{A}_1^1, \ldots, \mathbf{A}_{n_1}^1\} \times_\odot \{\mathbf{A}_1^2, \ldots, \mathbf{A}_{n_2}^2\} \times_\odot \cdots \times_\odot \{\mathbf{A}_1^r, \ldots, \mathbf{A}_{n_r}^r\}$. Then function

$$f_{n_1, \ldots, n_r}^F(x^1, \ldots, x^r) = \sum_{i_1=1}^{n_1} \cdots \sum_{i_r=1}^{n_r} F_{i_1 \cdots i_r} \mathbf{A}_{i_1}^1(x^1) \cdots \mathbf{A}_{i_r}^r(x^r) \tag{2}$$

is called the *inverse F-transform* of f.

3 Influence Variables Summarization

Now lets try to summarize meaning of variables which may play role in estimation of computational complexity:

- r – dimension of the original function domain, or in other words number of input variables.
- $n_1, \ldots, n_r$ – numbers of basic functions in particular dimensions. Lets denote $n = \bigvee_{i=1}^{r} n_i$.
- $\nu = n_1 \cdot n_2 \cdots n_r$ – the total number of components of fuzzy transform.
- N – count of sample nodes in which the original function is known or simply number of input data vectors (each of size $r+1$). The total size of input is then $M = N(r+1)$.

4 Algorithms

4.1 Discrete F-Transform

Algorithm of discrete fuzzy transform can be schematically described in the following way:

input : N sample points $p_i \in \mathcal{D}^r$
output: ν-tuple $[F_{i_1 \cdots i_r}]$ of real components

```
// initialization
```
setZeros(S^{fA});
setZeros(S^{A});

```
// main loop
```
foreach input data vector p **do**
    `// update value of all affected components`
    foreach combination $\mathbf{A}^1_{i_1}, \ldots \mathbf{A}^r_{i_r}$ of basic functions containing p **do**
        `// update auxiliary values of corresponding component` $F_{i_1 \cdots i_r}$
        $S^{fA}_{i_1 \cdots i_r} := S^{fA}_{i_1 \cdots i_r} + f(p^1, \ldots, p^r)\mathbf{A}^1_{i_1}(p^1) \cdots \mathbf{A}^r_{i_r}(p^r)$;
        $S^{A}_{i_1 \cdots i_r} := S^{A}_{i_1 \cdots i_r} + \mathbf{A}^1_{i_1}(p^1) \cdots \mathbf{A}^r_{i_r}(p^r)$;
    end
end

```
// compute value of all components
```
foreach component $F_{i_1 \cdots i_r}$ **do**
    $F_{i_1 \cdots i_r} := S^{fA}_{i_1 \cdots i_r} / S^{A}_{i_1 \cdots i_r}$;
end

Computational complexity of this algorithm can be divided into sum of three parts: initialization, main loop and finalization. Initialization and finalization parts both have obviously computational complexity in $o(n^r)$, where $n = \bigvee_{i=1}^{r} n_i$. Main loop will be iterated N times and contains nested loop which will be iterated 2^r times and complexity of its body is in $o(r)$. Computational complexity of Finding the first combination of basic functions $\mathbf{A}^1_{i_1}, \ldots \mathbf{A}^r_{i_r}$ depends on the type of partitioning. For uniform partitions it is in $o(r)$, but generally it is in $o(r \log n)$. So computational complexity of main loop is in $o(Nr2^r)$ for uniform partitions and generally in $o(Nr(2^r + \log n))$. Finally, the computational complexity of whole algorithm is consequently in $o(n^r + Nr2^r)$ for uniform partitions and in $o(n^r + Nr(2^r + \log n))$ for general partitions.

The last thing we may consider is the fact, that number of basic functions should be much less than number of sample points, so that $n^r \ll N$. In this case we can reduce previous estimations to $o(Nr2^r)$ for uniform partitions and $o(Nr(2^r + \log n))$ for general partitions.

4.2 Inverse Discrete F-Transform

Algorithm of inverse discrete fuzzy transform can be written in the following way:

```
input  : point p ∈ D^r
output: value of f^F_{n_1,...,n_r}(p^1, ..., p^r)
result := 0;
foreach combination A^1_{i_1}, ... A^r_{i_r} of basic functions containing p do
    result := result + F_{i_1···i_r} A^1_{i_1}(p^1) ··· A^r_{i_r}(p^r);
end
```

Computational complexity of inverse discrete f-transform algorithm consists only from complexity of one loop which is iterated 2^r times likewise as in the previous case. Computational complexity of Finding the first combination of basic functions $\mathbf{A}^1_{i_1}, \ldots \mathbf{A}^r_{i_r}$ depends on the type of partitioning as before. For uniform partitions it is in $o(r)$, but generally it is in $o(r \log n)$. Complexity of loop body is in $o(d)$, so that computational complexity of whole algorithm is in $o(r2^r)$ for uniform partitions and in $r(2^r + \log n)$ for general type of partitions.

5 Conclusion

It was shown that computational complexity of both direct and inverse fuzzy transforms depends on chosen type of partitions. When uniform partitions in all dimensions are used, both algorithms are more effective and their time consumption requirements are in $o(Nr2^r)$ for direct and $o(r2^r)$ for inverse f-transform. For general type of partitions the results are $o(Nr(2^r + \log n))$ for direct and $r(2^r + \log n)$ for inverse f-transform.

References

[1] Perfilieva, I., Khaldeeva, E.: *Fuzzy Transformation.* 9th IFSA World Congress & 20th NAFIPS Int. Conf. 20010725-20010728 Vancouver, Canada. Vancouver : IEEE, 2002. p. 1946-1948. ISBN 0-7803-7079-1.

[2] Perfilieva, I.: *Fuzzy Transforms.* Transactions on Rough Sets II. Fuzzy Sets and Rough Sets. Berlin : Springer-Verlag, 2004. ISBN 3-540-23990-1. p. 63-81.

[3] Dvořák, A., Habiballa, H., Novák, V., Pavliska, V.: *The software package LFLC 2000 – its specificity, recent and perspective applications.* Computers in Industry. 03. rel. 2003, vol. 51, p. 269 – 280

[4] Pavliska, V.: *Fuzzy Transform C++ Object Library.* Technical report No. 11, IRAFM, University of Ostrava, 2006.

[5] Štěpnička, M., Valášek, R.: *Fuzzy transforms and their application to wave equation.* Journal of Electrical Engineering, 55(12/s), pp. 7-10, 2004