



Sparse and Robust Signal Reconstruction

Sandra V. B. Jardim*

Tomar Polytechnic Institute, Department of Information Technology, Quinta do Contador, Estrada da Serra - Tomar, Portugal.

Abstract

Many problems in signal processing and statistical inference are based on finding a sparse solution to an undetermined linear system. The reference approach to this problem of finding sparse signal representations, on overcomplete dictionaries, leads to convex unconstrained optimization problems, with a quadratic term ℓ_2 , for the adjustment to the observed signal, and a coefficient vector ℓ_1 -norm. This work focus the development and experimental analysis of an algorithm for the solution of ℓ_q - ℓ_p optimization problems, where $p \in]0, 1] \wedge q \in [1, 2]$, of which ℓ_2 - ℓ_1 is an instance. The developed algorithm belongs to the majorization-minimization class, where the solution of the problem is given by the minimization of a progression of majorizers of the original function. Each iteration corresponds to the solution of an ℓ_2 - ℓ_1 problem, solved by the projected gradient algorithm. When tested on synthetic data and image reconstruction problems, the results shows a good performance of the implemented algorithm, both in compressed sensing and signal restoration scenarios.

Keywords: Sparse signal representation, Convex relaxation, ℓ_2 - ℓ_1 optimization, Compressed sensing, Majorization-minimization algorithms, Quadratic programming, Gradient projection algorithms.
2010 MSC: 93C42, 94A12.

1. Introduction

In general, sparse approximation problems have been of great interest given its wide applicability both in signal and image processing field as in statistical inference contexts, where many of the problems to be solved involve the undetermined linear systems sparse solutions determination.

The literature on sparsity optimization is rapidly increasing (see (Zarzer, 2009; Donoho, 2006; Candes & Tao, 2005; Wright *et al.*, 2009) and references therein). More recently sparsity techniques are also receiving increased attention in the optimal control community (Stadler, 2009; Casas *et al.*, 2012; Herzog *et al.*, 2012).

Given an input signal \mathbf{y} , sparse approximation problems resolution aims an approximated signal determination \mathbf{x} through a linear combination of elementary signals, which are, for several

*Corresponding author

Email address: sandra.jardim@ipt.pt (Sandra V. B. Jardim)

current applications, extracted from a set of signals not necessarily linearly independent. A preference for sparse linear combinations is imposed by penalizing nonzero coefficients. The most common penalty is the number of elementary signals that participate in the approximation. According to the context in which they operate and the objective to be achieved, sparse approximation problems can be formulated in different ways. In this sense, the problem domain must justify the linear model, the elementary signals choice and the sparsity criterion to be adopted.

On account of the combinatorial nature of the sparse approximation problems, which is due to the presence of the *quasi*-norm ℓ_0 of the coefficients vector to be estimated, these problems have a difficult computational resolution. In general, these optimization problems are NP-hard problems (Davis *et al.*, 1997; Natarajan, 1995). One of the most common approach to overcome this difficulty is the convex relaxation, introduced by Claerbout et al. (Claerbout & Muir, 1973), of the original problem, where the *quasi*-norm ℓ_0 is replaced by the norm ℓ_1 , which is a convex function. A classic example of this kind of problems is the determination of sparse representations on overcomplete dictionaries, where the reference approach leads to unconstrained convex optimization problems, which involves a quadratic term ℓ_2 of adjustment to the observed signal \mathbf{y} , and a ℓ_1 norm of the coefficient vector to be estimated \mathbf{x} . In this sense, it is notorious the interest shown by the scientific community in the development of methods leading to the resolution of the unconstrained convex optimization problem

$$\min_{\mathbf{x}} \frac{1}{2} \|\mathbf{y} - \phi \mathbf{x}\|_2^2 + \lambda \|\mathbf{x}\|_1, \quad (1.1)$$

where ϕ represents the overcomplete dictionary synthesis matrix; if $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{y} \in \mathbb{R}^k$, ϕ is a $k \times n$ matrix. The nonnegative parameter λ states a compromise between the approximation mean squared error and his sparsity level.

The above optimization problem, and related ones, arises in several applications, such as the *Basis Pursuit* and *Basis Pursuit Denoising* criterions (Chen *et al.*, 2001) and *Compressed Sensing* (Donoho, 2006).

The optimization problem represented in (1.1) is an instance of the general class of the optimization problems $\ell_q - \ell_p$, where q and p can assume values in the range $]0, 2]$. It is important to stress that the data term generalization to a ℓ_q norm, instead of the ℓ_2 norm, gives to the approximation criterion statistical strength features (when $q < 2$) (Huber & Ronchetti, 2009), making it less permeable to spurious observations (*outliers*). On the other hand, when it comes to generalization of the coefficient term to be estimated to a ℓ_p norm, instead a ℓ_1 norm, and considering $p < 1$, we walk toward the original combinatorial problem resolution.

In this paper is presented an algorithm that aims the resolution of the general class optimization problems $\ell_q - \ell_p$, where $p \in]0, 1] \wedge q \in [1, 2]$. The developed algorithm belong to the majorization-minimization class (Hunter & Lange, 2004), where the problem is solved in an iterative way, through the minimization of a majorizers sequence of the original function. Each upper bound function corresponds to a $\ell_2 - \ell_1$ problem, where each one of these problems is formulated as a quadratic programming problem and solved through the gradient projection algorithm (Figueiredo *et al.*, 2007b).

2. Generalized Optimization Problem

The unconstrained optimization problem represented in (1.1) is the convex relaxation of the subset selection problem, checking to be a major problem in many application areas. Due to its high applicability, there has been considerable effort made by the scientific community, regarding techniques and algorithms development for its resolution. Among the different proposed algorithms, are homotopy algorithms (Turlach, 2005; Efron *et al.*, 2004; Malioutov *et al.*, 2005), the ones based on interior-point methods (Turlach *et al.*, 2005; Chen *et al.*, 2001), the majorization-minimization class algorithms (Figueiredo & Nowak, 2003, 2005; Figueiredo *et al.*, 2007a) and the gradient projection algorithm (Figueiredo *et al.*, 2007b).

In this problem, the objective function composed by two terms, one of which being a quadratic term ℓ_2 of adjustment to the observed signal \mathbf{y} , and the other the ℓ_1 norm of the coefficient vector to estimate \mathbf{x} . Recall that the ℓ_1 norm arises by replacing the *quasi*-norm ℓ_0 , at the convex relaxation of the original convex optimization problem.

As stated above, the optimization problem $\ell_2 - \ell_1$ is an instance of the optimization problems $\ell_q - \ell_p$ general class, where $p \in]0, 1] \wedge q \in [1, 2]$.

Since the purpose of this work is to achieve the solution of the general class optimization problems $\ell_q - \ell_p$, for $p \in]0, 1] \wedge q \in [1, 2]$, let's consider the generalization of the unconstrained convex optimization problem (1.1), and define the function $L(\mathbf{x})$

$$L(\mathbf{x}) = \frac{1}{2} \|\mathbf{y} - \phi \mathbf{x}\|_q^q + \lambda \|\mathbf{x}\|_p^p, \quad (2.1)$$

where $\mathbf{x} \in \mathfrak{R}^n$, $\mathbf{y} \in \mathfrak{R}^k$, ϕ is the synthesis matrix of a dictionary D (of dimension $k \times n$), $\lambda \geq 0$, $p \in]0, 1]$ and $q \in [1, 2]$.

3. Majorization-Minimization Method for the Resolution of the Generalized Optimization Problem

3.1. Objective Function

There are several algorithms for the resolution of the optimization problem (1.1), such as Matching Pursuit (MP), Orthogonal Matching Pursuit (OMP), Homotopy, Least Absolute Shrinkage and Selection Operator (LASSO) and Gradient Projection (GPSR). In this work is developed a majorization-minimization (MM) method, for the resolution of the optimization problem $\min_{\mathbf{x}} L(\mathbf{x})$, where the optimization problem (1.1) is solved using the Barzilai-Borwein Gradient Projection algorithm for sparse reconstruction (GPSR-BB) (Figueiredo *et al.*, 2007b). This choice results from an analysis of the results obtained for different algorithms, which can be found in (Jardim, 2008). Since GPSR-BB algorithm aims to solve the optimization problem (1.1), it is necessary to establish a relation between this and the optimization problem that results from the minimization of the function $L(\mathbf{x})$ (2.1).

We can observe that the function $L(\mathbf{x})$ is the sum of the functions $L_1(\mathbf{x})$ and $L_2(\mathbf{x})$, where

$$L_1(\mathbf{x}) = \frac{1}{2} \|\mathbf{y} - \phi \mathbf{x}\|_q^q \quad \text{and} \quad (3.1)$$

$$L_2(\mathbf{x}) = \lambda \|\mathbf{x}\|_p^p. \tag{3.2}$$

Knowing that $\|w\|_r^r = \sum_i |w_i|^r$, we can define the function

$$f(z, p) = |z|^p. \tag{3.3}$$

Figures (1(a)) and (1(b)) are graphical representations of the function $f(z, p)$ for different values of p .

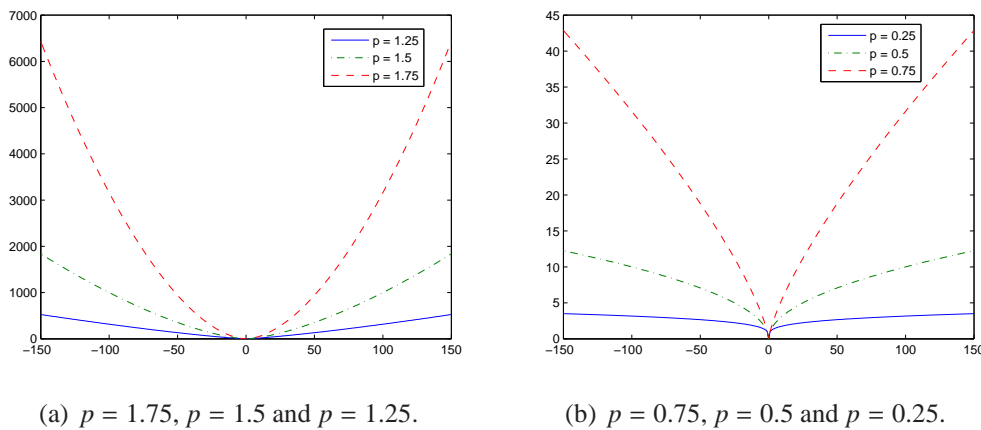


Figure 1. Function $f(z, p) = |z|^p$ for values of p greater and smaller than 1.

With this function it is possible to write (3.1) and (3.2) as

$$L_1(\mathbf{x}) = \frac{1}{2} \sum_{i=1}^k f(y_i - (\phi\mathbf{x})_i, q). \tag{3.4}$$

and

$$L_2(\mathbf{x}) = \lambda \sum_{i=1}^n f(x_i, p). \tag{3.5}$$

3.1.1. The Majorizer Function

By the analysis of the figures (1(a)) and (1(b)) we can verify that it is possible to determine for the function $f(\mathbf{z}, p)$ (so to the $L_1(\mathbf{x})$ and $L_2(\mathbf{x})$ functions), and for some value \mathbf{z}' , majorizers functions. This opens the door to the use of majorization-minimization algorithms to the resolution of the optimization problem $\min_{\mathbf{x}} L(\mathbf{x})$, where $L(\mathbf{x})$ is the function given by (2.1). So, it is necessary to define a Q function such that

$$L(\mathbf{x}) \leq Q(\mathbf{x}|\hat{\mathbf{x}}^{(t)}), \quad \forall_{\mathbf{x} \neq \hat{\mathbf{x}}^{(t)}} \tag{3.6}$$

$$L(\hat{\mathbf{x}}^{(t)}) = Q(\hat{\mathbf{x}}^{(t)}|\hat{\mathbf{x}}^{(t)}), \tag{3.7}$$

i.e., $Q(\mathbf{x}|\hat{\mathbf{x}}^{(t)})$ is a function of \mathbf{x} that majorizes (*i.e.*, upper bounds) $L(\mathbf{x})$.

Recalling that $L(\mathbf{x}) = L_1(\mathbf{x}) + L_2(\mathbf{x})$, where $L_1(\mathbf{x})$ and $L_2(\mathbf{x})$ are given by (3.4) and (3.5) respectively, let's consider the majorizer functions

$$Q_1(\mathbf{x}|\hat{\mathbf{x}}^{(t)}) \geq L_1(\mathbf{x}) \text{ and } Q_1(\hat{\mathbf{x}}^{(t)}|\hat{\mathbf{x}}^{(t)}) = L_1(\hat{\mathbf{x}}) \tag{3.8}$$

and

$$Q_2(\mathbf{x}|\hat{\mathbf{x}}^{(t)}) \geq L_2(\mathbf{x}) \text{ and } Q_2(\hat{\mathbf{x}}^{(t)}|\hat{\mathbf{x}}^{(t)}) = L_2(\hat{\mathbf{x}}^{(t)}). \tag{3.9}$$

Given the MM algorithm properties (Hunter & Lange, 2004), we can define a function

$$Q(\mathbf{x}|\hat{\mathbf{x}}^{(t)}) = Q_1(\mathbf{x}|\hat{\mathbf{x}}^{(t)}) + Q_2(\mathbf{x}|\hat{\mathbf{x}}^{(t)}) \text{ such that} \tag{3.10}$$

$$L(\mathbf{x}) \leq Q_1(\mathbf{x}|\hat{\mathbf{x}}^{(t)}) + Q_2(\mathbf{x}|\hat{\mathbf{x}}^{(t)}), \tag{3.11}$$

verifying

$$L(\hat{\mathbf{x}}^{(t)}) = Q_1(\hat{\mathbf{x}}^{(t)}|\hat{\mathbf{x}}^{(t)}) + Q_2(\hat{\mathbf{x}}^{(t)}|\hat{\mathbf{x}}^{(t)}). \tag{3.12}$$

Due to (3.4), and assuming q a fixed value in the interval $[1, 2]$, $L_1(\mathbf{x})$ is an even and growing function with $|\mathbf{y} - \phi \mathbf{x}|$ (see figure (1(a))), having a slower growth, or equal case $q = 2$, than a quadratic function of $|\mathbf{y} - \phi \mathbf{x}|$. So, it makes sense to use as majorizer function of $L_1(\mathbf{x})$ a quadratic function, which is also an even function, so without a linear term. In other words, we can use as $L_1(\mathbf{x})$ majorizer a function of the form $\frac{1}{2} \sum_i a_i (\mathbf{y} - \phi \mathbf{x})_i^2 + b_i$. The upper bound function can be used in a MM algorithm only if it verifies the key property, *i. e.*, the upper bound function must touch the function at the previous estimative. So it is necessary to determine a_i and b_i in order to

$$a_i (\mathbf{y} - \phi \mathbf{x})_i^2 + b_i \geq |(\mathbf{y} - \phi \mathbf{x})_i|^q, \forall \mathbf{x} \tag{3.13}$$

$$a_i (\mathbf{y} - \phi \mathbf{x}')_i^2 + b_i = |(\mathbf{y} - \phi \mathbf{x}')_i|^q. \tag{3.14}$$

Let's consider the function $f(z, q)$ given by (3.3), and a quadratic majorizer such that

$$g_1(z, z') = a z^2 + b \text{ such that:} \\ g_1(z, z') \geq f(z, q) \forall z \text{ and } g_1(z', z') = f(z', q). \tag{3.15}$$

Given that $f(z, q) = |z|^q$ we have, for $q \in [1, 2]$ and $z \neq 0$,

$$\frac{df(z, q)}{dz} = q|z|^{(q-1)} \text{sign}(z). \tag{3.16}$$

In order to $g_1(z, z')$ be tangent to $f(z, q)$ at $z = z'$, we get

$$a = \frac{q}{2}|z'|^{(q-2)}. \quad (3.17)$$

Since we also want $f(z', q) = g_1(z', z')$, we have

$$b = \frac{2-q}{2}|z'|^q. \quad (3.18)$$

Finally, we can write the majorizer function

$$g_1(z, z') = \frac{q}{2}(z')^{(q-2)}z^2 + \frac{2-q}{2}|z'|^q. \quad (3.19)$$

Based on majorizer (3.19) we can write

$$\begin{aligned} \frac{1}{2}\|\mathbf{y} - \phi \mathbf{x}\|_q^q &= \frac{1}{2} \sum_i |(\mathbf{y} - \phi \mathbf{x})_i|^q \\ &\leq \frac{q}{4} \sum_i |(\mathbf{y} - \phi \mathbf{x}')_i|^{(q-2)} (\mathbf{y} - \phi \mathbf{x})_i^2 \\ &\quad + \frac{2-q}{2} |(\mathbf{y} - \phi \mathbf{x}')_i|^q. \end{aligned} \quad (3.20)$$

Defining

$$\alpha_i^{(t)} = \frac{q}{2} \left(y_i - \sum_j (\phi_{ij} \hat{x}_j^{(t)}) \right)^{q-2}, \quad (3.21)$$

and

$$\beta_i^{(t)} = \frac{2-q}{2} \left| y_i - \sum_j (\phi_{ij} \hat{x}_j^{(t)}) \right|^q, \quad (3.22)$$

we have the majorizer $Q_1(\mathbf{x}|\hat{\mathbf{x}}^{(t)})$ given by

$$Q_1(\mathbf{x}|\hat{\mathbf{x}}^{(t)}) = \frac{1}{2} \sum_i \left[\alpha_i^{(t)} \left(y_i - \sum_j \phi_{ij} x_j \right)^2 + \beta_i^{(t)} \right]. \quad (3.23)$$

Since $L(\mathbf{x})$ is a function consisting of two separable terms, for which can be defined different majorizer functions, the majorizer function for the term $L_2(\mathbf{x}) = \lambda \|\mathbf{x}\|_p^p$ does not necessarily have to be a quadratic one. In fact, what is desirable is that the majorizer to be adopt for the $L_2(\mathbf{x})$ function, when added to the majorizer defined for $L_1(\mathbf{x})$, leads to a function $Q(\mathbf{x}, \hat{\mathbf{x}}^{(t)})$ with a minimizer easy to find. So, a ℓ_1 majorizer is the natural choice for penalties ℓ_p , with $0 < p \leq 1$, since it is more tighter than a quadratic majorizer. Recalling that, for the upper bound function can be used in a MM algorithm it must touch the function at the previous estimate, it is necessary to determine the parameters c and d so that

$$|x|^p \leq c|x| + d, \forall x \text{ and } |x'|^p \leq c|x'| + d.$$

Let's consider the function $f(x, p)$ given by (3.3), and a majorizer ℓ_1 :

$$\begin{aligned} g_2(x, x') &= c|x| + d \text{ such that:} \\ g_2(x, x') &\geq f(x, p), \forall x \text{ and } g_2(x', x') = f(x', p). \end{aligned} \quad (3.24)$$

Given that, for $p \in]0, 1]$, and $x' \neq 0$

$$\frac{df(x, p)}{dx} \Big|_{x=x'} = \text{sign}(x')p|x'|^{(p-1)}, \quad (3.25)$$

we have

$$c = p|x'|^{(p-1)}, \quad (3.26)$$

and

$$d = (1 - p)|x'|^p. \quad (3.27)$$

We can then write

$$g_2(x, x') = p|x'|^{(p-1)}|x| + (1 - p)|x'|^p. \quad (3.28)$$

Defining

$$\delta_i^{(t)} = \begin{cases} p|\hat{x}_i^{(t)}|^{(p-1)}, & \text{if } \hat{x}_i^{(t)} \neq 0 \\ 1/eps^a, & \text{if } \hat{x}_i^{(t)} = 0 \end{cases} \quad (3.29)$$

and

$$\epsilon_i^{(t)} = (1 - p)|\hat{x}_i^{(t)}|^p. \quad (3.30)$$

we have that the function $Q_2(\mathbf{x}|\hat{\mathbf{x}}^{(t)})$ can be written as

$$Q_2(\mathbf{x}|\hat{\mathbf{x}}^{(t)}) = \lambda \sum_i \left[\delta_i^{(t)} |x_i| + \epsilon_i^{(t)} \right]. \quad (3.31)$$

Given (3.10) we have that

$$\begin{aligned} Q(\mathbf{x}|\hat{\mathbf{x}}^{(t)}) &= \frac{1}{2} \sum_{i=1}^k \left[\alpha_i^{(t)} \left(y_i - \sum_j \phi_{ij} x_j \right)^2 + \beta_i^{(t)} \right] \\ &+ \lambda \sum_{i=1}^n \left[\delta_i^{(t)} |x_i| + \epsilon_i^{(t)} \right]. \end{aligned} \quad (3.32)$$

^a eps is the distance from 1.0 to the next larger double precision number, that is eps with no arguments returns $2^{(-52)}$.

Since at each step of the MM algorithm, we perform a minimization in \mathbf{x} , the terms $\beta_i^{(t)}$ and $\epsilon_i^{(t)}$ can be ignored, so we have

$$Q(\mathbf{x}|\hat{\mathbf{x}}^{(t)}) = \frac{1}{2} \sum_i \left[\alpha_i^{(t)} \left(y_i - \sum_j \phi_{ij} x_j \right)^2 \right] + \lambda \sum_i [\delta_i^{(t)} |x_i|], \quad (3.33)$$

or, in vectorial notation

$$Q(\mathbf{x}|\hat{\mathbf{x}}^{(t)}) = \frac{1}{2} (\mathbf{y} - \phi \mathbf{x})^T \Gamma^{(t)} (\mathbf{y} - \phi \mathbf{x}) + \lambda \mathbf{1}^T \Lambda^{(t)} |\mathbf{x}|, \quad (3.34)$$

with

$$\Gamma^{(t)} = \begin{bmatrix} \alpha_1^{(t)} & 0 & \dots & 0 \\ 0 & \alpha_2^{(t)} & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & \alpha_k^{(t)} \end{bmatrix}$$

and

$$\Lambda^{(t)} = \begin{bmatrix} \delta_1^{(t)} & 0 & \dots & 0 \\ 0 & \delta_2^{(t)} & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & \delta_n^{(t)} \end{bmatrix}. \quad (3.35)$$

and where $|\mathbf{x}| = [|\mathbf{x}_1|, |\mathbf{x}_2|, \dots, |\mathbf{x}_n|]^T$.

The minimization of the function $L(\mathbf{x})$ is implemented, iteratively, as a succession of minimalizations of the function $Q(\mathbf{x}|\hat{\mathbf{x}}^{(t)})$.

$$\begin{aligned} \hat{\mathbf{x}}^{(t+1)} &= \arg \min_{\mathbf{x}} Q(\mathbf{x}|\hat{\mathbf{x}}^{(t)}) \\ \hat{\mathbf{x}}^{(t+1)} &= \arg \min_{\mathbf{x}} \frac{1}{2} (\mathbf{y} - \phi \mathbf{x})^T \Gamma^{(t)} (\mathbf{y} - \phi \mathbf{x}) + \lambda \mathbf{1}^T \Lambda^{(t)} |\mathbf{x}|. \end{aligned} \quad (3.36)$$

Greater detail in the deduction of the presented mathematical expressions can be found in (Jardim, 2008).

4. Majorization-Minimization Algorithm

The minimization of the function $Q(\mathbf{x}|\hat{\mathbf{x}}^{(t)})$ (3.33) reflects an unconstrained convex optimization problem. In order to solve this problem with the algorithm GPSR-BB (Figueiredo *et al.*, 2007b),

it is necessary to reformulate the optimization problem (3.36) as a quadratic program (Nocedal & Wright, 1999), which leads to

$$\min_{\mathbf{z}} \mathbf{c}^T \mathbf{z} + \frac{1}{2} \mathbf{z}^T \mathbf{B} \mathbf{z} \tag{4.1}$$

subject to: $\mathbf{z} \geq 0$,

where $\mathbf{z} = [\mathbf{u}^T, \mathbf{v}^T]^T$ is a vector of unknown variables, with $\mathbf{u} = \max \{0, \mathbf{x}\}$ and $\mathbf{v} = \max \{0, -\mathbf{x}\}$, $\mathbf{c} = \lambda \mathbf{1}_{2n} + [-\mathbf{b}^T, \mathbf{b}^T]$ with $\mathbf{b} = \mathbf{A}^T \mathbf{y}$ and $\mathbf{A} = \phi^T \Gamma$, and $\mathbf{B} = \begin{bmatrix} \mathbf{A}^T \mathbf{A} & -\mathbf{A}^T \mathbf{A} \\ -\mathbf{A}^T \mathbf{A} & \mathbf{A}^T \mathbf{A} \end{bmatrix}$.

Considering the previously stated, the following algorithm was implemented in order to solve the optimization problem (4.1).

Step 0 (initialization): Given an initial estimate $\mathbf{z}^{(0)}$, set $t = 0$.

Step 1: Compute $\Gamma^{(t)}$ and $\tau^{(t)}$ according to (3.35) and $\tau_i = \lambda \Lambda_{ii}^{(t)}$, $\forall_{i=1, \dots, n}$, respectively.

Step 2: Execute GPSR-BB algorithm with entries the current estimate $\hat{\mathbf{x}}^{(t)}$, the $\tau^{(t)}$ vector, $\mathbf{A} = \Delta^{(t)} \phi$, and $\mathbf{y}_m = \Delta^{(t)} \mathbf{y}$.

$$\hat{\mathbf{x}}^{(t+1)} = \text{GPSR-BB}(\mathbf{y}_m, \mathbf{A}, \tau^{(t)}, \hat{\mathbf{x}}^{(t)}).$$

Step 3: Perform convergence test and terminate with approximated solution $\hat{\mathbf{x}}^{(t+1)}$; otherwise set $t = t + 1$ and return to **Step 1**.

4.1. Stopping Criterion

Initially we used the general stopping criterion

$$\frac{\|\hat{\mathbf{x}}^{(t+1)} - \hat{\mathbf{x}}^{(t)}\|_2}{\|\hat{\mathbf{x}}^{(t)}\|_2} \leq \varepsilon, \tag{4.2}$$

and it was found that it leads to good results. After that we choose a stopping criterion more directed to the problem to be solved, adopting the one it was used in the GPSR-BB algorithm, where the algorithm stops when the relative change in the number of nonzero components of the estimate falls below a given bound value.

4.2. Computational cost analysis

The number of iterations required to find an approximate solution, both in the outer cycle as in the inner one (**Step 2**), is not possible to accurately predict, since it depends (among other factors) on the quality of the initial estimate $\hat{\mathbf{x}}^{(0)}$. However, is possible to analyze the computational cost of each iteration of the proposed algorithm. For outer cycle, each iteration computational cost is essentially the inherent in the calculation of the matrix $\Gamma^{(t)}$, vectors $\tau^{(t)}$ and \mathbf{y}_m . The computation of the matrix $\Gamma^{(t)}$ matrix, as well as the vector \mathbf{y}_m , implies a matrix-vector product. To compute $\Gamma^{(t)}$ it is necessary to multiply the $(k \times n)$ matrix ϕ , by the vector of estimates $\hat{\mathbf{x}}^{(t)}$, of dimension n . This operation has a cost of $O(kn)$. To compute the vector \mathbf{y}_m is necessary to multiply the matrix $\Gamma^{(t)}$, of dimension $(k \times k)$, by the vector \mathbf{y} , of dimension k , which has a computational cost of $O(k)$. Calculate the vector $\tau^{(t)}$ implies vector-scalar product, which requires n floating-point operations.

The main computational cost per each cycle iteration of the GPSR-BB algorithm is a small number of inner products, scalar-vector multiplications, and vectors additions, each one of them requiring n or $2n$ floating-point operations, , plus a modest number of multiplications by \mathbf{A} and \mathbf{A}^T . The algorithm proposed in this paper for the resolution of the optimization problem that results from the minimization of the function $L(\mathbf{x})$ (2.1), executes GPSR-BB algorithm, where the matrix \mathbf{A} results from the product of the matrices ϕ and $\Gamma^{(t)}$. Given that ϕ is a $k \times n$ matrix, the computational cost of direct implementation of matrix-vector products by ϕ or ϕ^T is $O(kn)$. For $\Gamma^{(t)}$, $k \times k$ matrix, the computational cost of direct implementation of matrix-vector products is $O(k)$. If $\phi = \mathbf{R}\mathbf{W}$ is a matrix of dimension $k \times n$ and \mathbf{R} a $k \times d$ matrix, then \mathbf{W} must be a $d \times n$ matrix. If \mathbf{W} contains an orthogonal wavelet basis ($d = n$), matrix-vector products involving \mathbf{W} or \mathbf{W}^T can be implemented using fast wavelet tranform algorithms with $O(n)$ cost (Mallat, 1999), instead of the $O(n^2)$ cost of a direct matrix-vector product. Consequently, the cost of a product by ϕ or ϕ^T is $O(n)$ plus that of multiplying by \mathbf{R} or \mathbf{R}^T which, with a direct implementation, is $O(kn)$.

4.3. Convergence analysis

In order to analyze the convergence of the algorithm proposed in this paper, first will be analyzed the convergence of the GPSR-BB algorithm used in each iteration of the majorization-minimization algorithm, whose entries differ from those of the original algorithm of Figueiredo, being given by the equations defined above. Secondly will be analyzed the convergence of the iterative algorithm defined by the update (3.36).

As stated by Figueiredo in (Figueiredo *et al.*, 2007b) the convergence of the algorithm GPSR-BB used in this work can be derived from the analysis of Bertsekas (Bertsekas, 1999), but follows most directly from the results of Serafini, Zanghirati, and Zanni (Serafini *et al.*, 2005). In the algorithm proposed in this work we use the GPSR-BB algorithm with entries different from the ones defined by Figueiredo (Figueiredo *et al.*, 2007b). To summarize convergence properties of the GPSR-BB algorithm with the entries previously defined, we assume that termination occurs only when $\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)}$, which indicates that $\mathbf{x}^{(t)}$ is optimal.

Theorem 1: When $p = 1 \wedge q \in [1, 2]$ the sequence of iterates generated by the GPSR-BB algorithm with the entries \mathbf{y}_m , \mathbf{A} , $\tau^{(t)}$ and the current estimate $\hat{\mathbf{x}}^{(t)}$ either terminates at a solution of (4.1) or else converges to a solution of (4.1) at an R-linear rate.

Proof. Theorem 2.1 of (Serafini *et al.*, 2005) can be used to demonstrate that all accumulation points of $\{\mathbf{x}^{(t)}\}$ are stationary points. Although, in (Serafini *et al.*, 2005), this result applies to an algorithm in which the steplength parameters $\alpha^{(t)}$ used in the gradient projection method are chosen by a different scheme, the only relevant requirement on these parameters in the proof of [(Serafini *et al.*, 2005), Theorem 2.1] is that they lie in the range $[\alpha_{\min}, \alpha_{\max}]$, as in the case of the GPSR-BB algorithm used in this work. When $p = 1 \wedge q \in [1, 2]$, the objective in (4.1) is convex and bounded below, we can apply [(Serafini *et al.*, 2005), Theorem 2.2] to deduce convergence to a solution of (4.1) at an R-linear rate. When $p < 1$, the objective function in (4.1) is nonconvex; thus it can not be guaranteed that the algorithm converges to a global optimum. Nevertheless, in practice, we have never observed any convergence problems: the results show that the proposed algorithm finds actually a minimum, which although may not be a global minimum corresponds to a good reconstruction of the signal. \square

Theorem 2: For $Q(\mathbf{x}, \mathbf{x}')$ a continuous function in $(\mathbf{x}, \mathbf{x}')$ and L a strictly convex function, the MM iteration sequence $\hat{\mathbf{x}}^{(t)}$ converges to the global minimum of L .

Proof. Knowing that $\hat{\mathbf{x}}^{(t+1)} = \arg \min_{\mathbf{x}} Q(\mathbf{x}|\hat{\mathbf{x}}^{(t)})$, and recalling that the majorizer function Q verifies the conditions given by (3.6) and (3.7), we have

$$L(\hat{\mathbf{x}}^{(t+1)}) \leq Q(\hat{\mathbf{x}}^{(t+1)}|\hat{\mathbf{x}}^{(t)}) \leq Q(\hat{\mathbf{x}}^{(t)}|\hat{\mathbf{x}}^{(t)}) = L(\hat{\mathbf{x}}^{(t)}), \quad (4.3)$$

where the left hand inequality follows from the definition of Q and the right hand inequality from the definition of $\hat{\mathbf{x}}^{(t+1)}$. The sequence $L(\hat{\mathbf{x}}^{(t)})$, for $t = 1, 2, \dots$, is, therefore, nonincreasing. Under mild conditions, namely that $Q(\mathbf{x}, \mathbf{x}')$ is continuous in $(\mathbf{x}, \mathbf{x}')$, all limit points of the MM sequence $L(\hat{\mathbf{x}}^{(t)})$ are stationary points of L , and $L(\hat{\mathbf{x}}^{(t)})$ converges monotonically to $L^* = L(\mathbf{x}^*)$, for some stationary point \mathbf{x} . If, additionally, L is strictly convex, $\hat{\mathbf{x}}^{(t)}$ converges to the global minimum of L . Proofs of these properties are similar to those of similar properties of the EM algorithm (Huber & Ronchetti, 2009). \square

5. Experiments

In this section are presented, analyzed and discussed the results obtained by the proposed algorithm in compressed sensing applications and in the reconstruction of sparse images or with sparse representations, where for each signal the algorithm is tested for different values of \mathbf{p} and \mathbf{q} . Parameter λ is hand-tuned for the best SNR improvement. For compressed sensing scenarios it is adjusted according to the expression $\lambda = 0.1 \|\phi^T \mathbf{y}\|_{\infty}$ (as suggested by Fuchs in (Fuchs, 2004)).

All the experiments reported in this section were obtained with MATLAB (MATLAB 7.0 R14) implementations of the algorithm described above. The computing platform is a standard personal computer with Intel(R) Core(TM) i7 CPU, 8 GB of RAM, and running Windows 7 operating system.

5.1. Compressed Sensing

We first consider a typical compressed sensing (CS) scenario, where the goal is to reconstruct a length- n sparse signal (in the canonical basis, thus $\mathbf{W} = \mathbf{I}$ and $d = n$) from k observations, where

$k \leq n$. The rows of the $k \times n$ observation matrix \mathbf{R} are unit-norm random vectors (of Gaussian components) in \mathcal{R}^n . Notice that, since $k \leq n$, the system $\mathbf{R}\mathbf{x} = \mathbf{y}$ is undetermined. In the first example, we take $n = 2^{11} = 2048$, $k = 2^8 = 256$, and generate \mathbf{y} by adding Laplacian noise (probability density function $f(x) = a e^{-\frac{|x|}{a}}$) with parameter $a = 0.01$ to $\mathbf{R}\mathbf{x}$ (figure (2(b))). The original sparse signal \mathbf{x} is generated by a mixture of a uniform distribution on $[1, 1]$ and a point mass at zero, with probabilities 0.01 and 0.99, respectively. As we can see in 2(a), the original signal is indeed sparse.

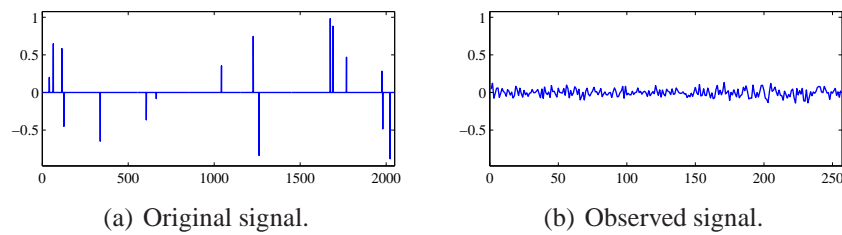


Figure 2. Original and observed signal.

For the described data set we have as initial estimate for $\ell_q - \ell_p$ algorithm the signal $\hat{\mathbf{x}}^{(0)} = \phi^T \mathbf{y}$. The estimates obtained by solving (2.1) using the proposed algorithm for $q = 1 \wedge p \in]0, 1]$ are shown in figure (3), and it can be verified that they are very close to the original signal.

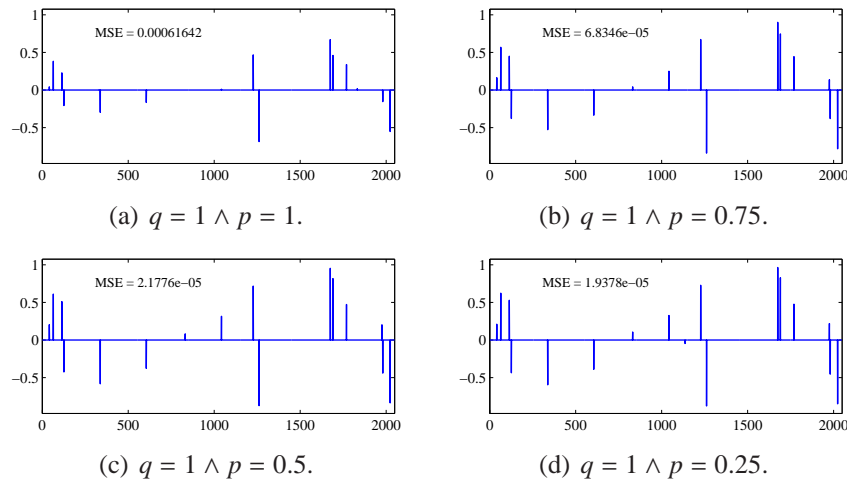


Figure 3. Estimated signal by (2.1) minimization for $q = 1 \wedge p \in]0, 1]$

From the results obtained, and presented in figure (3) it is possible to observe that the approximation mean squared error^b (MSE) for $q = 1 \wedge p = 1$ is about 10 times greater than that obtained for $p \leq 1$, meeting the expected results.

^bMSE = $(1/n) \|\hat{\mathbf{x}} - \mathbf{x}\|_2^2$, where $\hat{\mathbf{x}}$ is an estimate of \mathbf{x} .

As referred above, the generalization of the data term from a ℓ_2 norm to a ℓ_q norm, gives to the approximation criterion statistical strength features when $q < 2$. This can be verified in the second experiment, where we take $n = 2^{11} = 2048$, $k = 2^8 = 256$, and generate y by adding to $\mathbf{R}\mathbf{x}$ impulsive noise. Considering the original signal represented in figure (4(a)) with 16 nonzero components, we can see in figure (5) that the MSE of the approximation decreases with the value of q , taken as constant the value of $p = 1$.

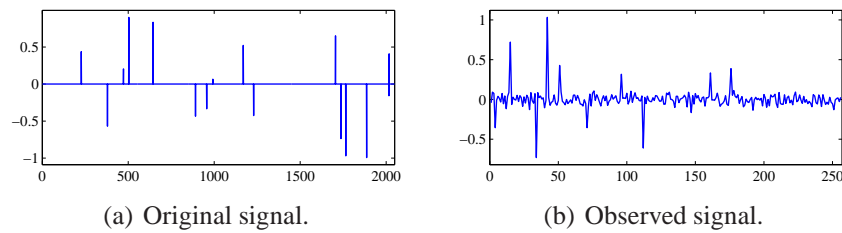


Figure 4. Original and observed signal.

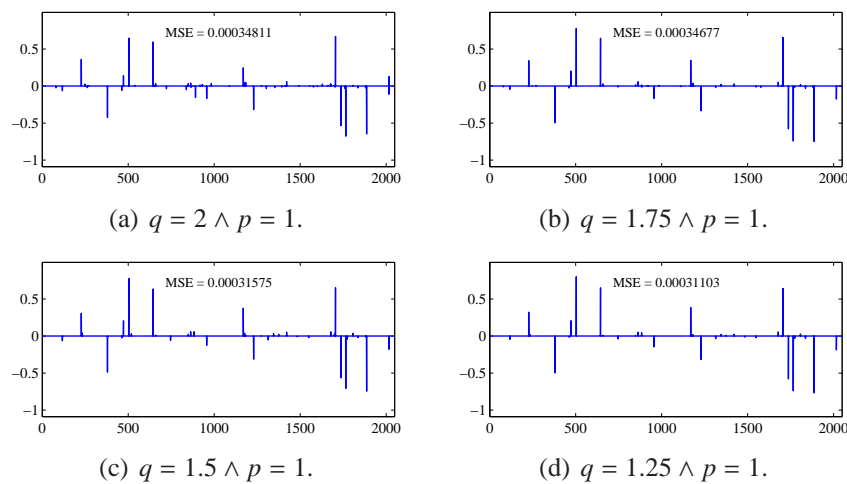


Figure 5. Estimated signal by (2.1) minimization for $q \in]1, 2] \wedge p = 1$

Note, for example, that the best result achieved by $\ell_q - \ell_p$ algorithm for $q = 2$ ($\text{MSE} = 3.4811 \times 10^{-4}$, 59 nonzero components) is not so good as the obtained for $q = 1.25$ ($\text{MSE} = 3.1103 \times 10^{-4}$, 35 nonzero components). Although a significant improvement in quantitative terms (the values of the approximations MSE for different values of q are not too different) doesn't occur, in this case, we can verify that, as the value of q decreases, the approach presents qualitative improvements (lower number of spurious observations in the final estimate). Figures (6(a)) and (6(b)) show the evolution of the MSE and objective function, for outer and inner loop respectively, against iteration number of the proposed algorithm, when $q = 1.5 \wedge p = 1$.

Analyzing the graphics of figure (6) we can observe that the values of the approximation MSE and the objective function decreases in each iteration of the $\ell_q - \ell_p$ algorithm. In fact, starting

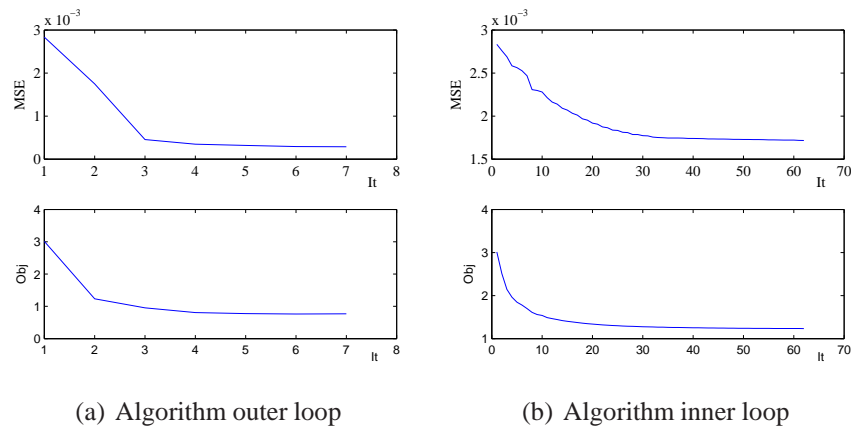


Figure 6. MSE and Objective function vs. Iteration number

from a initial upper bound function, built on the initial estimate, it is observed that the proposed algorithm builds at each iteration a new one, on the basis of the obtained solution for the previous upper bound function, which solution is closest to the original signal.

Next experiment shows the performance of the implemented algorithm in a typical compressed sensing problem, where the goal is to reconstruct a signal from k projections (with $k = 2^{10} = 1024$). The observation matrix \mathbf{R} , of dimension $k \times n$, is a matrix of random projection vectors. The two-dimensional original signal, which is represented in figure (7) has a sparse wavelet transform, and in this example the columns of the representation matrix \mathbf{W} form an orthogonal wavelet basis (daubechies-2 (Haar)). The original signal is an image of piecewise smooth filtered white noise of dimension $n' \times n'$, with $n' = 2^6$, *i. e.* $n = 2^{12}$ (figure (7)). The observed signal is obtained by adding white Gaussian noise with standard deviation 0.001 to $\mathbf{R}\mathbf{x}$. Figures (8(a)) - (8(c)) shows the results of $\ell_q - \ell_p$ algorithm, taken as initial estimate $\hat{\mathbf{x}}^{(0)} = 0$. By the results we can see that the proposed algorithm produces, from k projections corrupted by random white Gaussian noise best approximations (lower MSE) for $p < 1$.

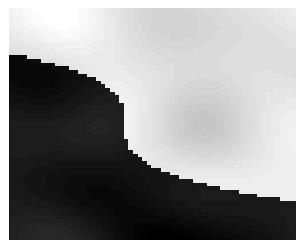


Figure 7. Original signal.

5.2. Image Restoration

The following two experiments show the performance of the proposed algorithm in real images restoration, where the columns of representation matrix \mathbf{W} form an orthogonal wavelet basis

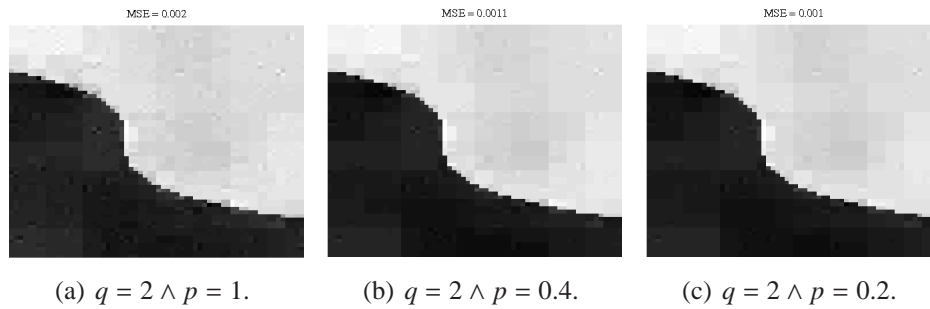


Figure 8. Estimated signal by (2.1) minimization, for $q = 2 \wedge p \in]0, 1]$.

(daubechies-2 (Haar)). The observation matrix \mathbf{R} , of dimension $k \times k$, is a Toeplitz blocks matrix representing 2D convolutions. In the first of these experiments we have as original signal the well-known Camera-man image (figure (9(a))). The observed image is obtained convolving the original image with a uniform blur filter of size 9×9 , and then adding to the blurred image ($\mathbf{R}\mathbf{x}$) white Gaussian noise with standard deviation 0.005 (figure (9(b))). Taken as initial estimate $\hat{\mathbf{x}}^{(0)} = \phi^T \mathbf{y}$, figures (10(a)) - (10(d)) shows the algorithm performance for $q = 2$ and $p \in]0, 1]$. For $q = 2 \wedge p = 0.5$ the initial value of the objective function is 2.0834×10^6 and the final one is 3.146×10^4 .

A typical statistical model for image wavelet coefficients is the Generalized Gaussian Density (Moulin & Liu, 1999), given by $p(\theta) = \exp\left\{-\frac{|\theta|^p}{\alpha}\right\}$, where θ represents the wavelet coefficients vector. The graph shown in figure (11) represents the evolution of MSE with p , where we can see that the best approximation occurs for $p = 0.5$. This is consistent with Moulin's statement (Moulin & Liu, 1999) that good image models based on wavelets are obtained doing $p \simeq 0.7$.



(a) Original image. (b) Observed image.

Figure 9. Original and observed real image.

In the last reported experiment the original image is the also well-known Lenna image (figure (12(a))). The observed image is obtained convolving the original image with a uniform blur filter of size 9×9 , and then adding to the blurred image ($\mathbf{R}\mathbf{x}$) impulsive noise (figure (12(b))).

Figures (13(a)) - (13(d)) shows the performance of the implemented algorithm for $q \in]1, 2]$ and $p = 1$. Again we can observe that best results are achieved for values of p below 1. For this experiment, and for $q = 1.25 \wedge p = 1$, we have that the initial value of the objective function is 3.7745×10^6 and the final one is 1.689×10^5 .

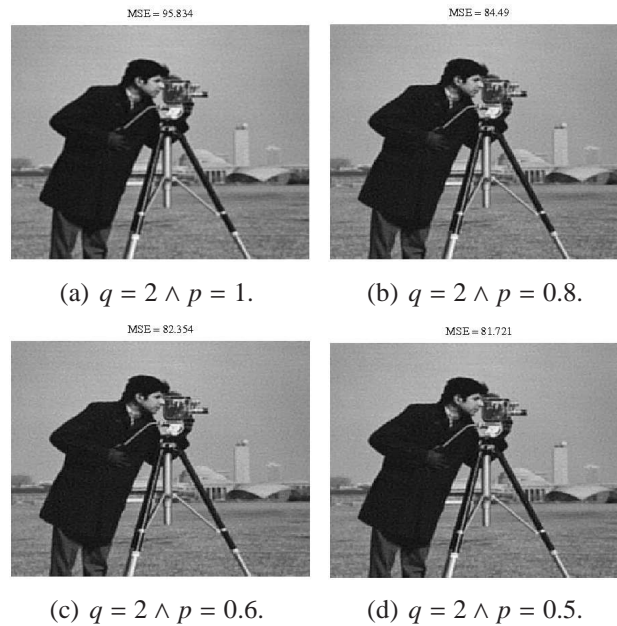


Figure 10. Estimated signal by (2.1) minimization, for $q = 2 \wedge p \in]0, 1]$.

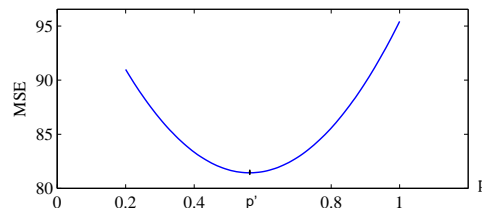


Figure 11. Approximation MSE evolution with p .

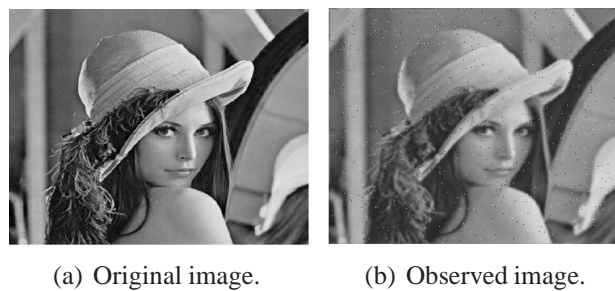


Figure 12. Original and observed real image.

6. Conclusions

In this paper was proposed a majorization-minimization class algorithm to address ℓ_q - ℓ_p optimization problems, where $p \in]0, 1] \wedge q \in [1, 2]$, of which ℓ_2 - ℓ_1 is an instance. The proposed algorithm was tested on scenarios of compressed sensing and image reconstruction, and, in both



Figure 13. Estimated signal by (2.1) minimization, for $q \in]1, 2]$ and $p = 1$.

cases, experiments were performed for data corrupted by uniform and impulsive noise. As mentioned, the generalization of the coefficients term to be estimated, from a ℓ_1 to a ℓ_p norm, with $p < 1$, aims to move towards the solution of the original combinatorial optimization problem, whose complexity prevents the calculation of a global solution in polynomial time. Although no one can guarantee the convergence of the algorithm to a global minimum, whereas for $p < 1$ the problem is no longer convex, the analysis of the results shows that the proposed algorithm provides as solution of the optimization problem a minimum corresponding to a signal reconstruction better than the one obtained by making $p = 1$ (lowest approximation MSE).

While in the experiments with nonnatural sparse signals we verify that approximation MSE, relatively to the original signal, decreases with the value of p , the same is not true for natural images, where we verify that there is an optimal value of $p < 1$, for which we obtain the best possible approximation, *i.e.*, that which corresponds the smallest MSE. This is justified against the model used for wavelet coefficients (GGD - Generalized Gaussian Density), which is function of the parameter p .

In the presence of outliers the proposed algorithm was tested taking p constant and equal to 1, and ranging q in $]1, 2]$. Both in compressed sensing applications of unidimensional signals, as in natural images restoration applications we can verify that the approximation MSE decreases with the value of parameter q . From the analysis of the results we can still observe that, as the value of q decreases the amount of outliers of the optimization problem solution also decreases. Hence it can be concluded that the use of ℓ_q norms, with $q < 2$, in data term, gives to the approximation criterion statistical robustness characteristics.

References

- Bertsekas, D. P. (1999). *Nonlinear Programming*. 2nd edition ed.. Athena Scientific.
- Candes, E. and T. Tao (2005). Decoding by linear programming. *IEEE Transactions on Information Theory* **51**(12), 4203–4215.
- Casas, E., C. Clason and K. Kunisch (2012). Approximation of elliptic control problems in measure spaces with sparse solutions. *SIAM Journal on Control and Optimization* **50**(4), 1735–1752.
- Chen, S., D. L. Donoho and M. A. Saunders (2001). Atomic decomposition by basis pursuit. *SIAM Journal on Scientific and Statistical Computing, Review* **43**, 129–159.
- Claerbout, J. and F. Muir (1973). Robust modelling of erratic data. *Geophysics* **38**, 826–844.
- Davis, G., S. Mallat and M. Avellaneda (1997). Adaptive greedy approximation. *Journal Constructive Approximations* **13**(1), 57–98.
- Donoho, D. L. (2006). Compressed sensing. *IEEE Transactions on Information Theory* **52**, 1289–1306.
- Efron, B., T. Hastie, I. Johnstone and R. Tibshirani (2004). Least angle regression. *The Annals of Statistics* **32**(2), 407–499.
- Figueiredo, M. A. T. and R. Nowak (2003). An em algorithm for wavelet-based image restoration. *IEEE Transactions on Image Processing* **12**, 906–916.
- Figueiredo, M. and R. Nowak (2005). A bound optimization approach to wavelet-based image deconvolution. In: *IEEE International Conference on Image Processing - ICIP'2005*.
- Figueiredo, M., J. Bioucas-Dias and R. Nowak (2007a). Majorization-minimization algorithms for wavelet-based image restoration. *IEEE Transactions on Image Processing: Special Issue on Convex Optimization Methods for Signal Processing* **16**(12), 2992–3004.
- Figueiredo, M., R. Nowak and S. Wright (2007b). Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems. *IEEE Journal of Selected Topics in Signal Processing: Special Issue on Convex Optimization Methods for Signal Processing* **1**(4), 586–598.
- Fuchs, J.-J. (2004). On sparse representations in arbitrary bases. *IEEE Transactions on Information Theory* **50**, 1341–1344.
- Herzog, R., G. Stadler and G. Wachsmuth (2012). Directional sparsity in optimal control of partial differential equations. *SIAM Journal on Control and Optimization* **50**(2), 943–963.
- Huber, P. J. and E. M. Ronchetti (2009). *Robust Statistics*. 2nd edition ed.. Wiley.
- Hunter, D. and K. Lange (2004). A tutorial on mm algorithms. *The American Statistician* **58**, 30–37.
- Jardim, S. (2008). Algoritmos para Representacao Esparsa e Robusta de Sinais. PhD thesis. Instituto Superior Tecnico, Universidade Tecnica de Lisboa. In Portuguese.
- Malioutov, D., M. Cetin and A. Willsky (2005). Homotopy continuation for sparse signal representation. In: *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*. Vol. 5. Philadelphia. pp. 733–736.
- Mallat, S. (1999). *A Wavelet Tour of Signal Processing*. 2nd edition ed.. Academic Press.
- Moulin, P. and J. Liu (1999). Analysis of multiresolution image denoising schemes using generalized-gaussian and complexity priors. *IEEE Transaction on Information Theory* **45**, 909–919.
- Natarajan, B. (1995). Sparse approximate solutions to linear systems. *SIAM Journal on Computing* **24**(2), 227–234.
- Nocedal, J. and S. Wright (1999). *Numerical Optimization*. Springer-Verlag. New York.
- Serafini, T., G. Zanghirati and L. Zanni (2005). Gradient projection methods for quadratic programs and applications in training support vector machines. *Optimization Methods and Software*.
- Stadler, G. (2009). Elliptic optimal control problems with l1-control cost and applications for the placement of control devices. *Computational Optimization and Applications* **44**(2), 159–181.

- Turlach, B. (2005). On algorithms for solving least squares problems under l_1 penalty or an l_1 constraint. In: *Proceedings of the American Statistical Association; Statistical Computing Section*. Alexandria. pp. 2572–2577.
- Turlach, B., W. N. Venables and S. J. Wright (2005). Simultaneous variable selection. *Technometrics* **27**, 349–363.
- Wright, S. J., R. D. Nowak and M. A. T. Figueiredo (2009). Sparse reconstruction by separable approximation. *IEEE Transactions on Signal Processing* **57**, 2479–2493.
- Zarzer, C. A. (2009). On tikhonov regularization with non-convex sparsity constraints. *Inverse Problems* **25:025006**, 13 pp.



Logical Tree of Mathematical Modeling

László Pokorádi*

Óbuda University Donát Bánki Faculty of Mechanical and Safety Engineering, 1081 Népszínház u. 8., Budapest, Hungary.

Abstract

During setting up a mathematical model, it can be very important and difficult task to choose input parameters that should be known for solution of this problem. A similar problem might come up when someone wants to carry out an engineering calculation task. A very essential aim technical education is developing of good logical engineering thinking. One main part of this thinking is to determine the potential sets of required input parameters of an engineering calculation. This paper proposes a logical tree based method to determine the required parameters of a mathematical model. The method gives a lively description about needed data base, and computational sequence for us to get to determine the set of required output parameter. The shown method is named **LogTreeMM - Logical Tree of Mathematical Modeling**.

Keywords: mathematical modeling, logical tree, engineering thinking, STEM education.
2010 MSC: 93A30, 00A71, 97D30.

1. Introduction

On the one hand, during engineering work, dozens and dozens of times we should determine any parameters of a technical system or process. To carry out the task mentioned above we have to know some input parameters of the investigated system or process. We can meet similar task during setting up a mathematical model for system or process simulation. The correct identification of optimal set of required input parameters is an important and hard engineering task.

On the other hand, during technical education it is a very difficult and essential task to develop good logical engineering thinking of students or pupils. One main part of this thinking is to determine the optimal set of required input parameters of the calculation task mentioned above.

The main aim of this paper is to show a logical tree method to determine required parameters of a mathematical model or an engineering calculation. This method gives a lively description about needed data base, and computational sequence for us to get to determine the required output

*Corresponding author

Email address: pokoradi.laszlo@bgk.uni-obuda.hu (László Pokorádi)

parameter. As the flow chart emphasizes the main steps of a calculation task, the proposed logical tree demonstrates the interdependencies and interrelations of variables. To choose a set of required parameters, firstly we should have an applicable equation to calculate the output parameter of the system that is a dependent variable of its model. Knowing this adaptable equation we can face the next question: How can we determine the independent variable(s) of the foregoing equation? And we should ask it repeatedly. ... It is possible that we can furnish two or more answers to one of these questions. In this case we get different required model parameter(s).

It is easily statable that we use logical inferences during determination of needed parameters to set up and to apply a mathematical model. In the one hand we use AND logical operations if all parameters should be known. On the other hand we use OR logical operations if we know two or more equations to calculate a parameter.

These logical connections are used in Fault Tree Analysis to determine causes of a system failure. The handbooks of NASA ([Stamatelatos & Caraballo, 2002](#)) and U.S. Nuclear Regulatory Commission ([Vesely et al., 1987](#)) show theoretical background and practical questions of FTA. There are several publications that propose reliability or safety methods based on FTA. For example, Tchorzewska-Cieslak and Boryczko presented the methodology of the FTA and an example of its application in order to analyze different failure scenarios in water distribution subsystem ([Tchorzewska-Cieslak & Boryczko, 2010](#)). They concluded that the FTA is particularly useful for the analysis of complex technical systems in which analysis of failure scenarios is a difficult process because it requires to examine a high number of cause-effect relationship. The water distribution subsystem undoubtedly belongs to such systems. The FTA involves thinking back, which allows the identification of failure events that cause the occurrence of the Top Event. In the case of very large fault trees it is advisable to use computer methods ([Tchorzewska-Cieslak & Boryczko, 2010](#)).

One of the FTA-based methods is the so-called bow-tie model that was used by Markowski and Kotynia ([Markowski & Kotynia, 2011](#)). It consists of a Fault Tree (FT) which identifies the causes of the undesired top event, and an Event Tree (ET) showing what is the consequence of such a release. So, this method encompasses the complete accident scenario using a bow-tie created by a Fault Tree and an Event Tree.

Pokorádi showed the adaptation of linear mathematical diagnostic modeling methodology for setting-up of Linear Fault Tree Sensitivity Model (LFTSM) ([Pokorádi, 2011](#)). The LFTSM is a modular approach tool that uses matrix-algebraic method based upon the mathematical diagnostic methodology of aircraft systems and gas turbine engines.

The logical method being presented in this paper is an adaptation of logical construction part of Fault Tree Analysis (FTA). This proposed method is named **LogTreeMM - Logical Tree of Mathematical Modeling**.

Pokorádi and Molnár showed the methodology of the Monte-Carlo Simulation and its applicability to investigate influences of fluid parameters to system losses by an easy pipeline system model. The (basically theoretical) obtained consequents and experiences can be used for investigation of parametrical uncertainties of the geothermal pipeline system, such as fluid characteristics indeterminations ([Pokorádi & Molnár, 2011](#)). This simulation model is practically used for demonstrating methodology of the proposed method.

The rest of this paper is organized as follows: Section 2 recalls the FTA methodology shortly

and the logical tree method theoretically. Section 3 presents a possibility of use of the proposed method by a case study. Section 4 summarizes the paper, outlines the prospective scientific work of the Author.

2. Theoretical delineation

The proposed method is an adaptation of logical construction of FTA. The FTA is a systematic, deductive (top-down type) and probabilistic risk assessment tool, which shows the causal relations leading to a given undesired event. Bell Telephone Laboratories developed its concept at the beginning of the 1960s. It was adopted later and extensively applied by Boeing Company. FTA is one of several symbolic "analytical logic techniques" found in operations research and in system reliability. The FTA is particularly useful for the analysis of complex technical systems where analysis of failure scenarios is a difficult process because it requires the examination of a high number of cause-effect relationships.

Fault Tree diagram displays on undesired state of the investigated system (Top Event) in terms of the states of its components (Basic Events). The FTA is a graphical design technique main result of which is a graph that has a dendritic structure.

The first step in a FTA is the selection of the Top Event that is a specific undesirable state or failure of a system.

After having analyzed the system so that we know all the causing effects we can construct the fault tree. Fault tree is based on **AND** and **OR** gates, which define the major characteristics of the fault tree.

The **AND** logical gate (Table 1) should be used if output event occurs only if all input events occur simultaneously. If the output event occurs if any of the input events occur, either alone or in any combinations, the **OR** logical gate (Table 1) should be used.

The Figure 1 shows a demonstrative Fault Tree. In the figure event B or C fail is Intermediate Event. The events A; B and C are Basic Events.

(After having the fault tree of the investigated undesired event, the probability of the Top Event can be analyzed depending of the probability of Basic Events. But it is not interesting in this study.)

The proposed method is analog with the above reviewed FTA technique. To construct LogTreeMM the following definitions are used:

A parameter can be known directly when

- its value is well-known (for example material characteristics);

or.

- it can be determined by direct measurement (for example internal diameter of a tube).

Let a variable be named Top Parameter if it should be determined but it is not used to calculate other one(s) in the investigated situation (is not an intermediate parameter).

Let a variable be called Intermediate Parameter if it has be known to calculate other one(s) but it cannot be known directly.

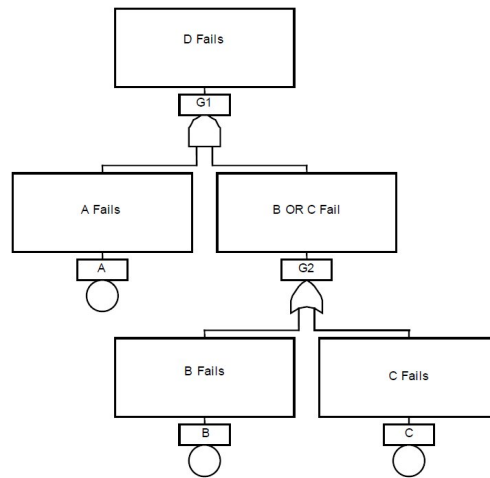







Figure 1. A Fault Tree (source: (Stamatelatos & Caraballo, 2002)).

Table 1

Symbols and Analogies between FTA and LogTreeMM		
Symbol	FTA	LogTreeMM
	Top Event	Top Parameter
	Intermediate Event	Intermediate Parameter
	Basic Event	Basic Parameter
	AND logical gate	
	OR logical gate	

Let a variable be named Base Parameter if

- it is known directly;

or

- it cannot be determined by any equation (relation).

Let an **AND** logical gate (Table 1) be used if all of the independent variables should be known to calculate a dependent variable of the given equation (relation).

Let an **OR** logical gate (Table 1) be used if there are more than one equation (relation) on even terms to calculate the given dependent variable.

Table 1 demonstrates symbols used in FTA and LogTreeMM, as well as the analogies between their events and gates.

3. Case Study

To demonstrate step by step the logical tree method introduced above, we show a case study based on the simulation model presented in [3]. The study aimed the investigation of influences of

fluid parameters to system losses in case of an easy pipeline system model. (Further on let the i -th logical gate be labeled by / i /.)

The illustrative system consisted of one lineal pipe and only one pipe fitting. The Δp pressure loss of this pipeline system as the Top Parameter can be determined by the equation

$$\Delta p = \Delta p_{cs} + \Delta p_{sz} \tag{3.1}$$

where:

Δp_{cs} pressure loss of linear pipe;

Δp_{sz} pressure loss of pipe fitting.

Thus two parameters (Δp_{cs} **AND** Δp_{sz}) should be known, which is shown by the **AND** gate of Figure 2. However, we do not know them directly. They are symbolized by rectangles, because they are Intermediate Elements of our investigation.

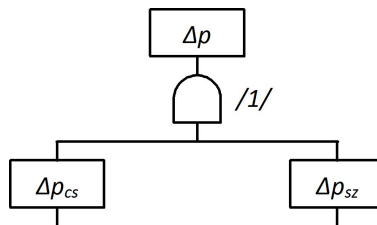


Figure 2. Logical Gate /1/ for Equation (3.1).

On the second level, the certain two structural elements (lineal pipe and pipe fitting) should be investigated. In the left branch, the pipe loss of linear pipe can be calculated by the equation

$$\Delta p_{cs} = \frac{\rho}{2} c^2 \frac{l}{d} \lambda \tag{3.2}$$

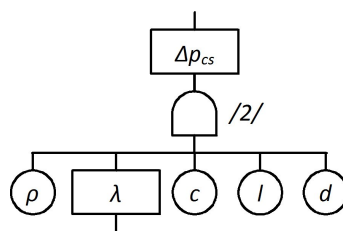


Figure 3. Logical Gate /2/ for Equation (3.2).

For that we should know fluid density ρ **AND** average fluid velocity c **AND** tube length l **AND** internal diameter d **AND** pipe loss coefficient λ . This logical sequence is shown by **AND** gate in Figure 3.

Except of the pipe lost coefficient (the rectangle shows that it is an Intermediate Parameter) all parameters can be determined directly (they are Basic Parameters), therefore they are shown by circles in Figure 3.

The pipe loss coefficient can be determined, depending only on Reynolds-number Re that cannot be determined directly (see Figure 4), by empirical equations in case of different Reynolds-number intervals:

$Re < 2320$:

$$\lambda = \frac{64}{Re} \tag{3.3}$$

$2320 < Re < 8 \cdot 10^4$:

$$\lambda = \frac{0.316}{\sqrt[4]{Re}} \tag{3.4}$$

$2 \cdot 10^4 < Re < 2 \cdot 10^6$:

$$\lambda = 0.0054 + 0.396 Re^{-0.3} \tag{3.5}$$

$10^5 < Re < 10^8$:

$$\lambda = 0.0032 + 0.211 Re^{-0.337} \tag{3.6}$$

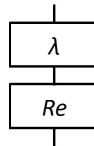


Figure 4. Connection of Equations (3.3) - (3.6).

For determination of the Reynolds-number Re the following equation should be used

$$Re = \frac{c d}{\nu} \tag{3.7}$$

For that we should know the average fluid velocity c **AND** the internal diameter d **AND** the kinematic viscosity of the fluid ν (they can be determined directly in other words, they are Basic Parameters).

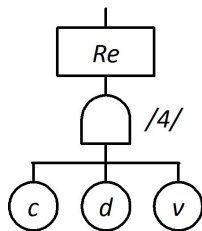


Figure 5. Logical Gate /4/ for Equation (3.7).

In the right branch of the second level, the loss pressure of pipe fitting can be determined by any of the following two equations

$$\Delta p_{sz} = \frac{\rho}{2} c^2 \xi \tag{3.8}$$

$$\Delta p_{sz} = \frac{\rho}{2} c^2 \frac{l_e}{d} \lambda \quad (3.9)$$

It is represented by the **OR** logical gate /3/ in Figure 6.

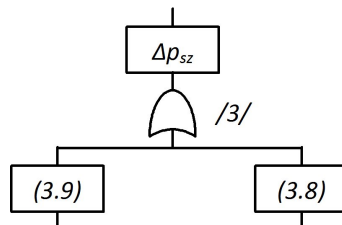


Figure 6. Logical Gate /3/.

In case of Equation (3.8), we need to know fluid density ρ **AND** average fluid velocity c **AND** pipe fitting loss coefficient ξ (see Figure 7).

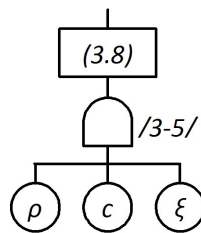


Figure 7. Logical Gate /3 – 5/ for Equation (3.8).

If the Equation (3.9) is used, the following parameters have to be known: fluid density ρ **AND** average fluid velocity c **AND** equivalent pipe length of pipe fitting l_e **AND** internal diameter of tube d **AND** pipe loss coefficient of tube λ (see Figure 8). The "equivalent pipe length" is length of given pipe, of which loss is equal to the loss of investigated pipe fitting in case of equal average fluid velocity.

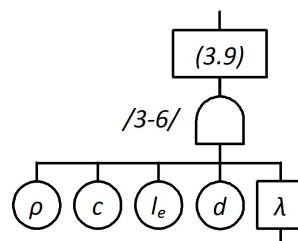


Figure 8. Logical Gate /3-6/ of Equation (3.9).

Assembling of the Figures 2 - 8, we get the logical tree of the illustrative model that is represented in Figure 9.

The sets of needed input parameters can be determined easily by investigating of the Logical Tree. For this purpose the subsets of the known parameters of the logical gates should be deduced first. In our case:

$$x_1 = \emptyset \tag{3.10}$$

$$x_2 = \{\rho; c; l; d\} \tag{3.11}$$

$$x_3 = \emptyset \tag{3.12}$$

$$x_4 = \{c; d; v\} \tag{3.13}$$

$$x_{3-5} = \{\rho; c; \xi\} \tag{3.14}$$

$$x_{3-6} = \{\rho; c; l_e; d\} \tag{3.15}$$

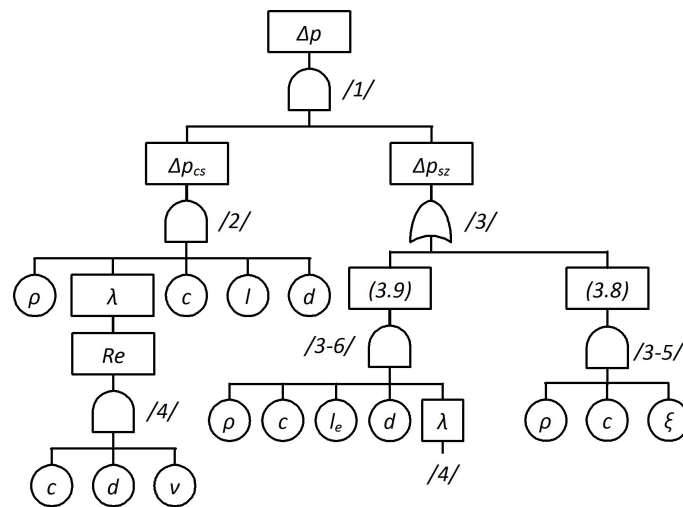


Figure 9. Logical Tree of the Case Study.

Knowing the subsets and the structure of the logical tree the possible sets of the needed parameters can be determined. The tree includes one two-input **OR** gate, therefore two sets of needed parameters can be identified:

$$x_A = x_2 \cup x_4 \cup x_{3-5} = \{\rho; c; l; d; v; \xi\} \tag{3.16}$$

$$x_A = x_2 \cup x_4 \cup x_{3-6} = \{\rho; c; l; d; v; l_e\} \tag{3.17}$$

It means that we should know either parameters from set x_A or from set x_B to set up and apply the mathematical model of the illustrative engineering problem. Fundamentally, the set x_A set was applied in the publication (Pokorádi & Molnár, 2011).

4. Conclusion

A logical tree method has been developed for the determination of possible sets of needed parameters for setting up of a mathematical model or solving an engineering calculation task. The method that named LogTreeMM is theoretically analogous with the Fault Tree Analysis used in system reliability assessment and quality management. The determined logical trees or their parts can be used as blocks to describe the required parameters in complex engineering calculation. In the education the LogTreeMM method can be used for developing of logical engineering thinking of students or pupils.

The Authors prospective scientific research related to this field of applied mathematics and engineering education includes the study of methodologies regarding technical system modeling and its decision making application in field of technical management.

References

- Markowski, Adam S. and Agata Kotynia (2011). "Bow-tie" model in layer of protection analysis. *Process Safety and Environmental Protection* **89**(4), 205 – 213.
- Pokorádi, L. (2011). Sensitivity investigation of fault tree analysis with matrix-algebraic method. *Theory and Applications of Mathematics and Computer Science* **1**(1), 34–44.
- Pokorádi, L. and B. Molnár (2011). Monte-Carlo simulation of the pipeline system to investigate water temperature's effects. *Polytechnical University of Bucharest. Scientific Bulletin. Series D: Mechanical Engineering* **73**(4), 223–236.
- Stamatelatos, M. and J. Caraballo (2002). *Fault Tree Handbook with Aerospace Applications, Office of safety and mission assurance NASA headquarters*. NASA: Washington DC.
- Tchorzewska-Cieslak, B. and K. Boryczko (2010). Relaxed LMI conditions for closed-loop fuzzy systems with tensor-product structure. *Engineering Applications of Artificial Intelligence* pp. 309–320.
- Vesely, W.E., Goldberg F.F., Norman R. and Haasl D. (1987). *Fault tree handbook, Government Printing Office*. Government Printing Office: Washington DC.



Luhn Prime Numbers

Octavian Cira^{a,*}, Florentin Smarandache^b

^a*Department of Mathematics and Computer Science, "Aurel Vlaicu" University of Arad, România.*

^b*Mathematics & Science Department, University of New Mexico, USA.*

Abstract

The first prime number with the special property that its addition with reversal gives as result a prime number too is 229. The prime numbers with this property will be called *Luhn prime numbers*. In this article we intend to present a performing algorithm for determining the *Luhn prime numbers*. Using the presented algorithm all the 50598 *Luhn prime numbers* have been, for p prime smaller than $2 \cdot 10^7$.

Keywords: Prime numbers, Reversal number, Smarandache's function, Luhn prime numbers.

2010 MSC: 11B83.

1. Introduction

The number 229 is the smallest prime number that added with his reverse gives as result a prime number, too. As $1151 = 229 + 922$ is prime.

The first that noted this special property the number 229 has, was Norman Luhn (after 9 February 1999), on the *Prime Curios* website (Caldwell & Honacher Jr., 2014). The prime numbers with this property will be later called *Luhn prime numbers*.

In the *Whats Special About This Number?* list (Friedman, 2014), a list that contains all the numbers between 1 and 9999; beside the number 229 is mentioned that his most important property is that, adding with reversal the resulting number is prime too.

The *On-Line Encyclopedia of Integer Sequences*, (Sloane, 2014, A061783), presents a list 1000 *Luhn prime numbers*. We owe this list to Harry J. Smith, since 28 July 2009. On the same website it is mentioned that Harvey P. Dale published on 27 November 2010 a list that contains 3000 *Luhn prime numbers* and Bruno Berselli published on 5 August 2013 a list that contains 2400 *Luhn prime numbers*.

*Corresponding author

Email addresses: octavian.cira@uav.ro (Octavian Cira), fsmarandache@gmail.com (Florentin Smarandache)

2. Smarandache’s function

The function $\mu : \mathbb{N}^* \rightarrow \mathbb{N}^*$, $\mu(n) = m$, where m is the smallest natural number with the property that n divides $m!$ (or $m!$ is a multiple of n) is known in the specialty literature as Smarandache’s function, (Smarandache, 1980, 1999; Sondow & Weisstein, 2014). The values resulting from $n = 1, 2, \dots, 18$ are: 1, 2, 3, 4, 5, 3, 7, 4, 6, 5, 11, 4, 13, 7, 5, 6, 17, 6. These values were obtained with an algorithm that results from μ ’s definition. The program using this algorithm cannot be used for $n \geq 19$ because the numbers $19!, 20!, \dots$ are numbers which exceed the 17 decimal digits limit and the classic computing model (without the arbitrary precisions arithmetic (Uznanski, 2014)) will generate errors due to the way numbers are represented in the computers memory.

3. Kempner’s algorithm

Kempner created an algorithm to calculate $\mu(n)$ using classical factorization $n = p_1^{p_1} \cdot p_2^{p_2} \cdot \dots \cdot p_s^{p_s}$, prime number and the generalized numeration base $(\alpha_i)_{[p_i]}$, for $i = \overline{1, s}$, (Kempner, 1918). Partial solutions to the algorithm for $\mu(n)$ ’s calculation have been given earlier by Lucas and Neuberg, (Sondow & Weisstein, 2014).

Remark. If $n \in \mathbb{N}^*$, n can be decomposed in a product of prime numbers $n = p_1^{\alpha_1} \cdot p_2^{\alpha_2} \cdot \dots \cdot p_s^{\alpha_s}$, where p_i are prime numbers so that $p_1 < p_2 < \dots < p_s$, and $s \geq 1$, thus Kempner’s algorithm for calculating the μ function is.

$$\mu(n) = \max \left\{ p_1 \cdot (\alpha_{1[p_1]})_{(p_1)}, p_2 \cdot (\alpha_{2[p_2]})_{(p_2)}, \dots, p_s \cdot (\alpha_{s[p_s]})_{(p_s)} \right\},$$

where by $(\alpha_{[p]})_{(p)}$ we understand that α is "written" in the numeration base p (noted $\alpha_{[p]}$) and it is "read" in the p numeration base (noted $\beta_{(p)}$, where $\beta = \alpha_{[p]}$), (Smarandache, 1999, p. 39).

4. Programs

The list of prime numbers was generated by a program that uses the Sieve of Eratosthenes the linear version of Pritchard, Pritchard (1987), which is the fastest algorithm to generate prime numbers until the limit of L , if $L \leq 10^8$. The list of prime numbers until to $2 \cdot 10^7$ is generated in about 5 seconds. For the limit $L > 10^8$ the fastest algorithm for generating the prime numbers is the Sieve of Atkin, Atkin & Bernstein (2004).

Program 4.1. *The Program for the Sieve of Eratosthenes, the linear version of Pritchard using minimal memory space is:*

```
CEPbm(L) :=
    λ ← floor(L/2)
    for k ∈ 1..λ
        is_prime_k ← 1
    prime ← (2 3 5 7)T
    i ← last(prime) + 1
    for j ∈ 4, 7..λ
        is_prime_j ← 0
```

```

k ← 3
s ← (primek-1)2
t ← (primek)2
while t ≤ L
    for j ∈ t, t + 2 · primek..L
        is_prime $\frac{j-1}{2}$  ← 0
    for j ∈ s + 2, s + 4..t - 2
        if is_prime $\frac{j-1}{2}$  = 1
            primei ← j
            i ← i + 1
    s ← t
    k ← k + 1
    t ← (primek)2
for j ∈ s + 2, s + 4..L
    if is_prime $\frac{j-1}{2}$  = 1
        primei ← j
        i ← i + 1
return prime
    
```

Program 4.2. The factorization program of a natural number; this program uses the vector p representing prime numbers, generated with the Sieve of Eratosthenes. The Sieve of Eratosthenes is called upon through the following sequence:

$$L := 2 \cdot 10^7 \quad t_0 = \text{time}(0) \quad p := \text{CEPbm}(L) \quad t_1 = \text{time}(1)$$

$$(t_1 - t_0)s = 5.064s \quad \text{last}(p) = 1270607 \quad p_{\text{last}(p)} = 19999999$$

```

Fa(m) := return ("m = " m " > ca ultimul p2") if m > (plast(p))2
    j ← 1
    k ← 0
    f ← (1 1)
    while m ≥ pj
        if mod(m, pj) = 0
            k ← k + 1
            m ←  $\frac{m}{p_j}$ 
        otherwise
            f ← stack[f, (pj, k)] if k > 0
            j ← j + 1
            k ← 0
    f ← stack[f, (pj, k)] if k > 0
    return submatrix(f, 2, rows(f), 1, 2)
    
```

We presented the Kempner’s algorithm using Mathcad programs required for the algorithm.

Program 4.3. The function counting all the digits in the p base of numeration in which is n .

$$ncb(n, p) := \begin{cases} \text{return } \text{ceil}(\log(n, p)) & \text{if } n > 1 \\ \text{return } 1 & \text{otherwise} \end{cases}$$

Where $\text{ceil}(x)$ is a Mathcad function which gives the smallest integer $\geq x$ and $\log(n, p)$ is logarithm in base p from n .

Program 4.4. The program intended to generate the p generalized base of numeration (noted by Smarandache $[p]$) for a number with m digits.

$$a(p, m) := \begin{cases} \text{for } i \in 1..m \\ \quad a_i \leftarrow \frac{p^i - 1}{p - 1} \\ \text{return } a \end{cases}$$

Program 4.5. The program intended to generate for the p base of numeration (noted by Smarandache (p)) to write the α number.

$$b(\alpha, p) := \begin{cases} \text{return } (1) & \text{if } p = 1 \\ \text{for } i \in 1..ncb(\alpha, p) \\ \quad b_i \leftarrow p^{i-1} \\ \text{return } b \end{cases}$$

Program 4.6. Program that determines the digits of the generalized base of numeration $[p]$ for the number n .

$$Nbg(n, p) := \begin{cases} m \leftarrow ncb(n, p) \\ a \leftarrow a(p, m) \\ \text{return } (1) & \text{if } m=0 \\ \text{for } i \in m..1 \\ \quad \begin{cases} c_i \leftarrow \text{trunc}\left(\frac{n}{a_i}\right) \\ n \leftarrow \text{mod}(n, a_i) \end{cases} \\ \text{return } c \end{cases}$$

Where $\text{trunc}(x)$ returns the integer part of x by removing the fractional part, and $\text{mod}(x, y)$ returns the remainder on dividing x by y (x modulo y).

Program 4.7. Program for Smarandache's function.

$$\mu(n) := \begin{cases} \text{return } \text{"Err. } n \text{ nu este intreg"} & \text{if } n \neq \text{trunc}(n) \\ \text{return } \text{"Err. } n < 1" & \text{if } n < 1 \\ \text{return } (1) & \text{if } n=1 \\ f \leftarrow Fa(n) \\ p \leftarrow f^{(1)} \\ \alpha \leftarrow f^{(2)} \\ \text{for } k = 1..rows(p) \\ \quad \eta_k \leftarrow p_k \cdot Nbg(\alpha_k, p_k) \cdot b(\alpha_k, p_k) \\ \text{return } \max(\eta) \end{cases}$$

This program calls the $Fa(n)$ factorization with prime numbers. The program uses the Smarandache's Remark 3 – about the Kempner algorithm. The $\mu.prn$ file generation is done once. The reading of this generated file in Mathcad's documents results in a great time–save.

Program 4.8. Program with which the file $\mu.prn$ is generated

$$VF\mu(N) := \left| \begin{array}{l} \mu_1 \leftarrow 1 \\ \text{for } n \in 2..N \\ \quad \mu_n \leftarrow \mu(n) \\ \text{return } \mu \end{array} \right.$$

This program calls the program 4.7 for calculating the value of the μ function. The sequence of the $\mu.prn$ file generation is:

$$t_0 := \text{time}(0) \quad \text{WRITEPRN}(\text{"}\mu.prn\text{"}) := VF\mu(2 \cdot 10^7) \quad t_1 := \text{time}(1)$$

$$(t_1 - t_0)\text{sec} = \text{"}5 : 17 : 32.625\text{"}hhmmss$$

Smarandache's function is important because it characterizes prime numbers – through the following fundamental property:

Theorem 4.9. Let be p an integer > 4 , than p is prime number if and only if $\mu(p) = p$.

Proof. See (Smarandache, 1999, p. 31). □

Hence, the fixed points of this function are prime numbers (to which is added 4). Due to this property the function is used as primality test.

Program 4.10. Program for testing μ 's primality. This program returns the 0 value if the number is not prime number and the 1 value if the number is a prime. The file $\mu.prn$ will be read and it will be assigned to the μ vector.

$$\text{ORIGIN} := 1 \quad \mu := \text{READPRN}(\text{"}\dots \backslash \mu.prn\text{"})$$

$$Tp\mu(n) := \left| \begin{array}{l} \text{return "Err. } n < 1 \text{ sau } n \notin \mathbb{Z}\text{" if } n < 1 \vee n \neq \text{trunc}(n) \\ \text{if } n > 4 \\ \quad \left| \begin{array}{l} \text{return } 0 \text{ if } \mu_n \neq n \\ \text{return } 1 \text{ otherwise} \end{array} \right. \\ \text{otherwise} \\ \quad \left| \begin{array}{l} \text{return } 0 \text{ if } n=1 \vee n=4 \\ \text{return } 1 \text{ otherwise} \end{array} \right. \end{array} \right.$$

Program 4.11. Program that provides the reverses of the given m number.

$$R(m) := \left| \begin{array}{l} n \leftarrow \text{floor}(\log(m)) \\ x \leftarrow m \cdot 10^{-n} \\ \text{for } k \in 1..n \\ \quad \left| \begin{array}{l} c_k \leftarrow \text{trunc}(x) \\ x \leftarrow (x - c_k) \cdot 10 \end{array} \right. \\ c_{n+1} \leftarrow \text{round}(x) \\ Rm \leftarrow 0 \\ \text{for } k \in n + 1..2 \\ \quad Rm \leftarrow (Rm + c_k) \cdot 10 \\ \text{return } Rm + c_1 \end{array} \right.$$

Where $\text{floor}(x)$ returns the greatest integer $\leq x$ and $\text{round}(x)$ returns x rounded to the nearest integer.

Program 4.12. Search program for the Luhn prime numbers.

$$PLuhn(L) := \left| \begin{array}{l} n \leftarrow \text{last}(p) \\ S \leftarrow (229) \\ k \leftarrow 51 \\ \text{while } p_k \leq L \\ \quad \left| \begin{array}{l} N \leftarrow R(p_k) + p_k \\ S \leftarrow \text{stack}(S, p_k) \text{ if } T p \mu(N) = 1 \\ k \leftarrow k + 1 \end{array} \right. \\ \text{return } S \end{array} \right.$$

The function $\text{stack}(A, B, \dots)$ is applied for merging matrixes top-down. The number of columns in matrixes should also be the same. The discussed functions could be applied to vectors as well.

Execution of the program $PLuhn$ was made with sequence

$$S := PLuhn(2 \cdot 10^7)$$

The initialization of the S stack is done with the vector that contains the number 229. The variable k has the initial value of 51 because the index of the 229 prime number is 50, so that the search for the *Luhn prime numbers* will begin with $p_{51} = 233$.

5. List of prime numbers Luhn

We present a partial list of the 50598 *Luhn prime numbers* smaller than $2 \cdot 10^7$ (the first 321 and the last 120):

229, 239, 241, 257, 269, 271, 277, 281, 439, 443, 463, 467, 479, 499, 613, 641, 653, 661, 673, 677, 683, 691, 811, 823, 839, 863, 881, 20011, 20029, 20047, 20051, 20101, 20161, 20201, 20249, 20269, 20347, 20389, 20399, 20441, 20477, 20479, 20507, 20521, 20611, 20627, 20717, 20759, 20809, 20879, 20887, 20897, 20981, 21001, 21019, 21089, 21157, 21169, 21211, 21377, 21379, 21419, 21467, 21491, 21521, 21529, 21559, 21569, 21577, 21601, 21611, 21617, 21647, 21661,

21701, 21727, 21751, 21767, 21817, 21841, 21851, 21859, 21881, 21961, 21991, 22027, 22031, 22039, 22079, 22091, 22147, 22159, 22171, 22229, 22247, 22291, 22367, 22369, 22397, 22409, 22469, 22481, 22501, 22511, 22549, 22567, 22571, 22637, 22651, 22669, 22699, 22717, 22739, 22741, 22807, 22859, 22871, 22877, 22961, 23017, 23021, 23029, 23081, 23087, 23099, 23131, 23189, 23197, 23279, 23357, 23369, 23417, 23447, 23459, 23497, 23509, 23539, 23549, 23557, 23561, 23627, 23689, 23747, 23761, 23831, 23857, 23879, 23899, 23971, 24007, 24019, 24071, 24077, 24091, 24121, 24151, 24179, 24181, 24229, 24359, 24379, 24407, 24419, 24439, 24481, 24499, 24517, 24547, 24551, 24631, 24799, 24821, 24847, 24851, 24889, 24979, 24989, 25031, 25057, 25097, 25111, 25117, 25121, 25169, 25171, 25189, 25219, 25261, 25339, 25349, 25367, 25409, 25439, 25469, 25471, 25537, 25541, 25621, 25639, 25741, 25799, 25801, 25819, 25841, 25847, 25931, 25939, 25951, 25969, 26021, 26107, 26111, 26119, 26161, 26189, 26209, 26249, 26251, 26339, 26357, 26417, 26459, 26479, 26489, 26591, 26627, 26681, 26701, 26717, 26731, 26801, 26849, 26921, 26959, 26981, 27011, 27059, 27061, 27077, 27109, 27179, 27239, 27241, 27271, 27277, 27281, 27329, 27407, 27409, 27431, 27449, 27457, 27479, 27481, 27509, 27581, 27617, 27691, 27779, 27791, 27809, 27817, 27827, 27901, 27919, 28001, 28019, 28027, 28031, 28051, 28111, 28229, 28307, 28309, 28319, 28409, 28439, 28447, 28571, 28597, 28607, 28661, 28697, 28711, 28751, 28759, 28807, 28817, 28879, 28901, 28909, 28921, 28949, 28961, 28979, 29009, 29017, 29021, 29027, 29101, 29129, 29131, 29137, 29167, 29191, 29221, 29251, 29327, 29389, 29411, 29429, 29437, 29501, 29587, 29629, 29671, 29741, 29759, 29819, 29867, 29989,

...

8990143, 8990209, 8990353, 8990441, 8990563, 8990791, 8990843, 8990881, 8990929, 8990981, 8991163, 8991223, 8991371, 8991379, 8991431, 8991529, 8991553, 8991613, 8991743, 8991989, 8992069, 8992091, 8992121, 8992153, 8992189, 8992199, 8992229, 8992259, 8992283, 8992483, 8992493, 8992549, 8992561, 8992631, 8992861, 8992993, 8993071, 8993249, 8993363, 8993401, 8993419, 8993443, 8993489, 8993563, 8993723, 8993749, 8993773, 8993861, 8993921, 8993951, 8994091, 8994109, 8994121, 8994169, 8994299, 8994463, 8994473, 8994563, 8994613, 8994721, 8994731, 8994859, 8994871, 8994943, 8995003, 8995069, 8995111, 8995451, 8995513, 8995751, 8995841, 8995939, 8996041, 8996131, 8996401, 8996521, 8996543, 8996651, 8996681, 8996759, 8996831, 8996833, 8996843, 8996863, 8996903, 8997059, 8997083, 8997101, 8997463, 8997529, 8997553, 8997671, 8997701, 8997871, 8997889, 8997931, 8997943, 8997979, 8998159, 8998261, 8998333, 8998373, 8998411, 8998643, 8998709, 8998813, 8998919, 8999099, 8999161, 8999183, 8999219, 8999311, 8999323, 8999339, 8999383, 8999651, 8999671, 8999761, 8999899, 8999981.

6. Conclusions

The list of all *Luhn prime numbers*, that totaled 50598 numbers, was determined within a time span of 54 seconds, on an Intel processor of 2.20 GHz.

References

- Atkin, A. O. L. and D. J. Bernstein (2004). Prime Sieves Using Binary Quadratic Forms. *Math. Comp.* **73**, 1023–1030.
Caldwell, Ch. K. and G. L. Honacher Jr. (2014). Prime Curios! The Dictionary of Prime Number Trivia.
Friedman, E. (2014). What's Special About This Number? From: Erich's Place.

Kempner, A. J. (1918). Miscellanea. *Amer. Math. Monthly* **25**, 201–210.

Pritchard, P. (1987). Linear prime number sieves: a family tree. *Sci. Comp. Prog.* **9**(1), 17–35.

Sloane, N. J. A (2014). Primes p such that $p + (p \text{ reversed})$ is also a prime. From: The On-Line Encyclopedia of Integer Sequences.

Smarandache, F. (1980). O nouă funcție în teoria analitică a numerelor. *An. Univ. Timișoara* **XVIII**(fasc. 1), 79–88.

Smarandache, F. (1999). *Asupra unor noi funcții în teoria numerelor*. Universitatea de Stat Moldova. Chișinău, Republica Moldova.

Sondow, J. and E. W. Weisstein (2014). Smarandache Function.

Uznanski, D. (2014). Arbitrary precision.



The Applicability of $\$$ -Calculus to Solve Some Turing Machine Undecidable Problems

Eugene Eberbach^{a,*}

^a*Department of Engineering and Science, Rensselaer Polytechnic Institute,
Hartford, 275 Windsor Street, CT 06120, United States.*

Abstract

The $\$$ -calculus process algebra for problem solving applies the cost performance measures to converge in finite time or in the limit to optimal solutions with minimal problem solving costs. The $\$$ -calculus belongs to superTuring models of computation. Its main goal is to provide the support to solve hard computational problems. It allows also to solve in the limit some undecidable problems. In the paper we demonstrate how to solve in the limit Turing Machine Halting Problem, to approximate the universal search algorithm, to decide diagonalization language, nontrivial properties of recursively enumerable languages, and how to solve Post Correspondence Problem and Busy Beaver Problem.

Keywords: problem solving, hypercomputation, expressiveness, superTuring models of computation, resource bounded computation, process algebras, $\$$ -calculus.

2010 MSC: primary classification: 68Qxx, 68Txx.

2012 CCS: primary classification: Theory of computation, mathematics of computing. Secondary classifications: models of computation, Turing machines, concurrency, process calculi, formal languages and automata theory, formalisms, automata over infinite objects, mathematical optimization, discrete optimization.

1. Introduction

In this paper, the expressiveness of the $\$$ -calculus process algebra of bounded rational agents (Eberbach, 1997, 2005a, 2006, 2007) is investigated. The $\$$ -calculus, presented in this paper, belongs to superTuring models of computation and provides a support to handle intractability and undecidability in problem solving. In the paper, we present the applicability of $\$$ -calculus to solve (in hypercomputational sense) some undecidable problems.

The paper is organized as follows. In section 2, we briefly recall some basic notions related to Turing machine problem solving and hypercomputation. In section 3, we outline the $\$$ -calculus

*Corresponding author

Email address: eberbe@rpi.edu (Eugene Eberbach)

process algebra of bounded rational agents. In section 4, we present the solution of the halting problem of Universal Turing Machine, and approximate solution of the universal search algorithm. In section 5, other TM unsolvable problems are investigated, including the diagonalization language, nontrivial properties of recursively enumerable languages, Post Correspondence Problem and Busy Beaver Problem. Section 6 contains conclusions.

2. Problem Solving in Turing Machines and Hypercomputation

Turing Machines (TMs) (Turing, 1937, 1939) and algorithms are two fundamental concepts of computer science and problem solving. Turing Machines describe the limits of problem solving using conventional recursive algorithms, and laid the foundation of current computer science in the 1960s.

Note that there are several other models of algorithms, called super-recursive algorithms, that can compute more than Turing Machines, using hypercomputational/superTuring models of computation (Burgin, 2004; Syropoulos, 2007). The battle between reductionists (believing in strong Church-Turing Thesis and “unsinkability” of Turing machine model) and remodelers (hyper-computationalists trying to develop new super-Turing models of computation for solution of Turing Machine undecidable problems) is not over, however shifting gradually in favor of hyper-computationalists (Aho, 2011; Cooper, 2012; Wegner *et al.*, 2012).

It turns out that (TM) *undecidable problems* cannot be solved by TMs and *intractable problems* are solvable, but require too many resources (e.g., steps or memory). For undecidable problems effective recipes do not exist - problems are called nonalgorithmic or nonrecursive. On the other hand, for intractable problems algorithms exist, but running them on a deterministic Turing Machine, requires an exponential amount of time (the number of elementary moves of the TM) as a function of the TM input.

We use the simplicity of the TM model to prove formally that there are specific problems (languages) that the TM cannot solve (Hopcroft *et al.*, 2001). Solving the problem is equivalent to decide whether a string belongs to the language. A problem that cannot be solved by computer (Turing machine) is called *undecidable* (TM-undecidable). The class of languages accepted by Turing machines are called *recursively enumerable (RE-) languages*. For RE-languages, TM can accept the strings in the language but cannot tell for certain that a string is not in the language.

There are two classes of Turing machine unsolvable languages (problems):

recursively enumerable RE but not recursive - TM can accept the strings in the language but cannot tell for certain that a string is not in the language (e.g., the language of the universal Turing machine, or Post’s Correspondence Problem language). A language is decidable but its complement is undecidable, or vice versa: a language is undecidable but its complement is decidable.

non-RE - no TM can even recognize the members of the language in the RE sense (e.g., the diagonalization language). Neither a language nor its complement is decidable.

Decidable problems have a (recursive) algorithm, i.e., TM halts whether or not it accepts its input. Decidable problems are described by *recursive languages*. Algorithms as we know are associated

with the class of recursive languages, a subset of recursively enumerable languages for which we can construct its accepting TM. For recursive languages, both a language and its complement are decidable.

Turing Machines are used as a formal model of classical (recursive) algorithms. An algorithm should consist of a finite number of steps, each having well defined and implementable meaning. We are convinced that computer computations are not restricted to such restrictive definition of algorithms only.

Definition 2.1. *By superTuring computation (also called hypercomputation) we mean any computation that cannot be carried out by a Turing Machine as well as any (algorithmic) computation carried out by a Turing Machine.*

In (Eberbach & Wegner, 2003; Eberbach *et al.*, 2004), several superTuring models have been discussed and overviewed. The incomplete list includes Turing’s o-machines, c-machines and u-machines, cellular automata, discrete and analog neural networks, Interaction Machines, Persistent Turing Machines, Site and Internet Machines, the π -calculus, the $\$$ -calculus, Inductive Turing Machines, Infinite Time Turing Machines, Accelerating Turing Machines and Evolutionary Turing Machines. In particular, the author proposed two superTuring models of computation: the $\$$ -Calculus (Eberbach, 2005a, 2007) and Evolutionary Turing Machine (Eberbach, 2005b; Eberbach & Burgin, 2009).

SuperTuring models derive their higher than the TM expressiveness using three principles: *interaction*, *evolution*, or *infinity*. In the *interaction principle* the model becomes open and the agent interacts with either a more expressive component or with an infinite many components. In the *evolution principle*, the model can evolve to a more expressive one using non-recursive variation operators. In the *infinity principle*, models can use unbounded resources: time, memory, the number of computational elements, an unbounded initial configuration, an infinite alphabet, etc. The details can be found in (Eberbach & Wegner, 2003; Eberbach *et al.*, 2004).

3. The $\$$ -Calculus Algebra of Bounded Rational Agents

The $\$$ -calculus is a mathematical model of processes capturing both the final outcome of problem solving as well as the interactive incremental way how the problems are solved. The $\$$ -calculus is a process algebra of Bounded Rational Agents for interactive problem solving targeting intractable and undecidable problems. It has been introduced in the late of 1990s (Eberbach, 1997, 2005a, 2007). The $\$$ -calculus (pronounced cost calculus) is a formalization of resource-bounded computation (also called anytime algorithms), proposed by Dean, Horvitz, Zilberstein and Russell in the late 1980s and early 1990s (Horvitz & Zilberstein, 2001; Russell & Norvig, 2002). Anytime algorithms are guaranteed to produce better results if more resources (e.g., time, memory) become available. The standard representative of process algebras, the π -calculus (Milner *et al.*, 1992; Milner, 1999) is believed to be the most mature approach for concurrent systems.

The $\$$ -calculus rests upon the primitive notion of *cost* in a similar way as the π -calculus was built around a central concept of *interaction*. Cost and interaction concepts are interrelated in the sense that cost captures the quality of an agent interaction with its environment. The unique feature of the $\$$ -calculus is that it provides a support for problem solving by incrementally searching for

solutions and using cost to direct its search. The basic $\$$ -calculus search method used for problem solving is called $k\Omega$ -optimization. The $k\Omega$ -optimization represents this “impossible” to construct, but “possible to approximate indefinitely” universal algorithm. It is a very general search method, allowing the simulation of many other search algorithms, including A*, minimax, dynamic programming, tabu search, or evolutionary algorithms. Each agent has its own Ω search space and its own limited horizon of deliberation with depth k and width b . Agents can cooperate by selecting actions with minimal costs, can compete if some of them minimize and some maximize costs, and be impartial (irrational or probabilistic) if they do not attempt to optimize (evolve, learn) from the point of view of the observer. It can be understood as another step in the never ending dream of universal problem solving methods recurring throughout all computer science history. The $\$$ -calculus is applicable to robotics, software agents, neural nets, and evolutionary computation. Potentially it could be used for design of cost languages, cellular evolvable cost-driven hardware, DNA-based computing and molecular biology, electronic commerce, and quantum computing. The $\$$ -calculus leads to a new programming paradigm *cost languages* and a new class of computer architectures *cost-driven computers*.

3.1. The $\$$ -Calculus Syntax

In $\$$ -calculus everything is a cost expression: agents, environment, communication, interaction links, inference engines, modified structures, data, code, and meta-code. $\$$ -expressions can be simple or composite. Simple $\$$ -expressions α are considered to be executed in one atomic indivisible step. Composite $\$$ -expressions P consist of distinguished components (simple or composite ones) and can be interrupted.

Definition 3.1. The $\$$ -calculus The set \mathcal{P} of $\$$ -calculus process expressions consists of simple $\$$ -expressions α and composite $\$$ -expressions P , and is defined by the following syntax:

α	::=	$(\$_{i \in I} P_i)$	cost
		$(\rightarrow_{i \in I} c P_i)$	send P_i with evaluation through channel c
		$(\leftarrow_{i \in I} c X_i)$	receive X_i from channel c
		$(\prime_{i \in I} P_i)$	suppress evaluation of P_i
		$(a_{i \in I} P_i)$	defined call of simple $\$$ -expression a with parameters P_i , and and its optional associated definition ($:= (a_{i \in I} X_i) < R >$) with body R evaluated atomically
		$(\bar{a}_{i \in I} P_i)$	negation of defined call of simple $\$$ -expression a
P	::=	$(\circ_{i \in I} \alpha P_i)$	sequential composition
		$(\parallel_{i \in I} P_i)$	parallel composition
		$(\cup_{i \in I} P_i)$	cost choice
		$(\cup_{i \in I} P_i)$	adversary choice
		$(\sqcup_{i \in I} P_i)$	general choice
		$(f_{i \in I} P_i)$	defined process call f with parameters P_i , and its associated definition ($:= (f_{i \in I} X_i) R$) with body R (normally suppressed); ($^1 R$) will force evaluation of R exactly once

The indexing set I is a possibly countably infinite. In the case when I is empty, we write empty parallel composition, general, cost and adversary choices as \perp (blocking), and empty sequential composition (I empty and $\alpha = \varepsilon$) as ε (invisible transparent action, which is used to mask, make invisible parts of $\$$ -expressions). Adaptation (evolution/upgrade) is an essential part of $\$$ -calculus, and all $\$$ -calculus operators are infinite (an indexing set I is unbounded). The $\$$ -calculus agents interact through send-receive pair as the essential primitives of the model.

Sequential composition is used when $\$$ -expressions are evaluated in a textual order. Parallel composition is used when expressions run in parallel and it picks a subset of non-blocked elements at random. Cost choice is used to select the cheapest alternative according to a cost metric. Adversary choice is used to select the most expensive alternative according to a cost metric. General choice picks one non-blocked element at random. General choice is different from cost and adversary choices. It uses guards satisfiability. Cost and adversary choices are based on cost functions. Call and definition encapsulate expressions in a more complex form (like procedure or function definitions in programming languages). In particular, they specify recursive or iterative repetition of $\$$ -expressions.

Simple cost expressions execute in one atomic step. Cost functions are used for optimization and adaptation. The user is free to define his/her own cost metrics. Send and receive perform handshaking message-passing communication, and inferencing. The suppression operator suppresses evaluation of the underlying $\$$ -expressions. Additionally, a user is free to define her/his own simple $\$$ -expressions, which may or may not be negated.

3.2. The $\$$ -Calculus Semantics: The $k\Omega$ -Search

In this section we define the operational semantics of the $\$$ -calculus using the $k\Omega$ -search that captures the dynamic nature and incomplete knowledge associated with the construction of the problem solving tree.

The basic $\$$ -calculus problem solving method, the $k\Omega$ -optimization, is a very general search method providing meta-control, and allowing to simulate many other search algorithms, including A^* , minimax, dynamic programming, tabu search, or evolutionary algorithms (Russell & Norvig, 2002). The problem solving works iteratively: through select, examine and execute phases. In the select phase the tree of possible solutions is generated up to k steps ahead, and agent identifies its alphabet of interest for optimization Ω . This means that the tree of solutions may be incomplete in width and depth (to deal with complexity). However, incomplete (missing) parts of the tree are modeled by silent $\$$ -expressions ε , and their cost estimated (i.e., not all information is lost). The above means that $k\Omega$ -optimization may be if some conditions are satisfied to be complete and optimal. In the examine phase the trees of possible solutions are pruned minimizing cost of solutions, and in the execute phase up to n instructions are executed. Moreover, because the $\$$ operator may capture not only the cost of solutions, but the cost of resources used to find a solution, we obtain a powerful tool to avoid methods that are too costly, i.e., the $\$$ -calculus directly minimizes search cost. This basic feature, inherited from anytime algorithms, is needed to tackle directly hard optimization problems, and allows to solve total optimization problems (the best quality solutions with minimal search costs). The variable k refers to the limited horizon for optimization, necessary due to the unpredictable dynamic nature of the environment. The variable Ω refers to a reduced alphabet of information. No agent ever has reliable information about all factors that

influence all agents behavior. To compensate for this, we mask factors where information is not available from consideration; reducing the alphabet of variables used by the $\$$ -function. By using the $k\Omega$ -optimization to find the strategy with the lowest $\$$ -function, meta-system finds a satisficing solution, and sometimes the optimal one. This avoids wasting time trying to optimize behavior beyond the foreseeable future. It also limits consideration to those issues where relevant information is available. Thus the $k\Omega$ optimization provides a flexible approach to local and/or global optimization in time or space. Technically this is done by replacing parts of $\$$ -expressions with invisible $\$$ -expressions ε , which remove part of the world from consideration (however, they are not ignored entirely - the cost of invisible actions is estimated).

The $k\Omega$ -optimization meta-search procedure can be used both for single and multiple cooperative or competitive agents working online ($n \neq 0$) or offline ($n = 0$). The $\$$ -calculus programs consist of multiple $\$$ -expressions for several agents.

Let's define several auxiliary notions used in the $k\Omega$ -optimization meta-search. Let:

- \mathcal{A} - be an alphabet of $\$$ -expression names for an enumerable *universe of agent population* (including an environment, i.e., one agent may represent an environment). Let $\mathcal{A} = \bigcup_i A_i$, where A_i is the alphabet of $\$$ -expression names (simple or complex) used by the i -th agent, $i = 1, 2, \dots, \infty$. We will assume that the names of $\$$ -expressions are unique, i.e., $A_i \cap A_j = \emptyset, i \neq j$ (this always can be satisfied by indexing $\$$ -expression name by a unique agent index. This is needed for an agent to execute only own actions). The agent population size will be denoted by $p = 1, 2, \dots, \infty$.
- $x_i[0] \in \mathcal{P}$ - be an initial $\$$ -expression for the i -th agent, and its initial search procedure $k\Omega_i[0]$.
- $\min(\$_i(k\Omega_i[t], x_i[t]))$ - be an implicit default goal and $Q_i \subseteq \mathcal{P}$ be an optional (explicit) goal. The default goal is to find a pair of $\$$ -expressions, i.e., any pair $(k\Omega_i[t], x_i[t])$ being

$$\min\{\$ _i(k\Omega_i[t], x_i[t]) = \$_{1i}(\$_{2i}(k\Omega_i[t]), \$_{3i}(x_i[t]))\},$$

where $\$_{3i}$ is a problem-specific cost function, $\$_{2i}$ is a search algorithm cost function, and $\$_{1i}$ is an aggregating function combining $\$_{2i}$ and $\$_{3i}$. This is the default goal for total optimization looking for the best solutions $x_i[t]$ with minimal search costs $k\Omega_i[t]$. It is also possible to look for the optimal solution only, i.e., the best $x_i[t]$ with minimal value of $\$_{3i}$, or the best search algorithm $k\Omega_i[t]$ with minimal costs of $\$_{2i}$. The default goal can be overwritten or supplemented by any other termination condition (in the form of an arbitrary $\$$ -expression Q) like the maximum number of iterations, the lack of progress, etc.

- $\$ _i$ - a cost function performance measure (selected from the library or user defined). It consists of the problem specific cost function $\$_{3i}$, a search algorithm cost function $\$_{2i}$, and an aggregating function $\$_{1i}$. Typically, a user provides cost of simple $\$$ -expressions or an agent can learn such costs (e.g., by reinforcement learning). The user selects or defines also how the costs of composite $\$$ -expressions will be computed. The cost of the solution tree is the function of its components: costs of nodes (states) and edges (actions). This allows to express both the quality of solutions and search cost.

- $\Omega_i \subseteq \mathcal{A}$ - a *scope of deliberation/interests of the i -th agent*, i.e., a subset of the universe's of \mathcal{A} -expressions chosen for optimization. All elements of $\mathcal{A} - \Omega_i$ represent irrelevant or unreachable parts of an environment, of a given agent or other agents, and will become invisible (replaced by ε), thus either ignored or unreachable for a given agent (makes optimization local spatially). Expressions over $A_i - \Omega_i$ will be treated as observationally congruent (cost of ε will be neutral in optimization, e.g., typically set to 0). All expressions over $\Omega_i - A_i$ will be treated as strongly congruent - they will be replaced by ε and although invisible, their cost will be estimated using the best available knowledge of an agent (may take arbitrary values from the cost function domain).
- $b_i = 0, 1, 2, \dots, \infty$ - a branching factor of the search tree (LTS), i.e., the maximum number of generated children for a parent node. For example, hill climbing has $b_i = 1$, for binary tree $b_i = 2$, and $b_i = \infty$ is a shorthand to mean to generate all children (possibly infinite many).
- $k_i = 0, 1, 2, \dots, \infty$ - represents *the depth of deliberation*, i.e., the number of steps in the derivation tree selected for optimization in the examine phase (decreasing k_i prevents combinatorial explosion, but can make optimization local in time). $k_i = \infty$ is a shorthand to mean to the end to reach a goal (may not require infinite number of steps). $k_i = 0$ means omitting optimization (i.e., the empty deliberation) leading to reactive behaviors. Similarly, a branching factor $b_i = 0$ will lead to an empty deliberation too. Steps consist of multi-sets of simple \mathcal{A} -expressions, i.e., a parallel execution of one or more simple \mathcal{A} -expressions constitutes one elementary step.
- $n_i = 0, 1, 2, \dots, \infty$ - the number of steps selected for execution in the execute phase. For $n_i > k_i$ steps larger than k_i will be executed without optimization in reactive manner. For $n_i = 0$ execution will be postponed until the goal will be reached.
For the depth of deliberation $k_i = 0$, the $k\Omega$ -search will work in the style of imperative programs (reactive agents), executing up to n_i consecutive steps in each loop iteration. For $n_i = 0$ search will be offline, otherwise for $n_i \neq 0$ - online.
- *gp, reinf, strongcon, update* - auxiliary flags used in the $k\Omega$ -optimization meta-search procedure.

Each agent has its own $k\Omega$ -search procedure $k\Omega_i[t]$ used to build the solution $x_i[t]$ that takes into account other agent actions (by selecting its alphabet of interests Ω_i that takes actions of other agents into account). Thus each agent will construct its own view of the whole universe that only sometimes will be the same for all agents (this is an analogy to the subjective view of the “objective” world by individuals having possibly different goals and different perception of the universe).

Definition 3.2. The $k\Omega$ -Optimization Meta-Search Procedure The $k\Omega$ - optimization meta-search procedure $k\Omega_i[t]$ for the i -th agent, $i = 0, 1, 2, \dots$, from an enumerable universe of agent population and working in time generations $t = 0, 1, 2, \dots$ is a complex \mathcal{A} -expression (meta-procedure) consisting of simple \mathcal{A} -expressions $init_i[t]$, $sel_i[t]$, $exam_i[t]$, $goal_i[t]$, $\mathcal{S}_i[t]$, complex \mathcal{A} -expression $loop_i[t]$ and $exec_i[t]$, and constructing solutions, its input $x_i[t]$, from predefined and user defined simple

and complex $\$$ -expressions. For simplicity, we will skip time and agent indices in most cases if it does not cause confusion, and we will write *init*, *loop*, *sel*, *exam*, $goal_i$ and $\$i$. Each i -th agent performs the following $k\Omega$ -search procedure $k\Omega_i[t]$ in the time generations $t = 0, 1, 2, \dots$:

```
(:= (kΩi[t] xi[t]) (◦ (init (kΩi[0] xi[0])) // initialize kΩi[0] and xi[0]
(loop xi[t + 1])) // basic cycle: select, examine,
) // execute
```

where *loop* meta- $\$$ -expression takes the form of the select-examine-execute cycle performing the $k\Omega$ -optimization until the goal is satisfied. At that point, the agent re-initializes and works on a new goal in the style of the never ending reactive program:

```
(:= (loop xi[t]) // loop recursive definition
(⊔ (◦ (goali[t] (kΩi[t] xi[t])) // goal not satisfied, default goal
// min($i (kΩi[t] xi[t]))
(sel xi[t]) // select: build problem solution tree k step
// deep, b wide
(exam xi[t]) // examine: prune problem solution tree in
// cost ∪ and in adversary ∪ choices
(exec (kΩi[t] xi[t])) // execute: run optimal xi n steps and
// update kΩi parameters
(loop xi[t + 1])) // return back to loop
(◦ (goali[t] (kΩi[t] xi[t])) // goal satisfied - re-initialize search
(kΩi[t] xi[t])))
)
```

Simple $\$$ -expressions *init*, *sel*, *exam*, *goal* with their atomically executed bodies are defined below. On the other hand, *exec* can be interrupted after each action, thus it is not atomic.

1. **Initialization** ($:=$ (*init* ($k\Omega_i[0]$ $x_i[0]$)) $<$ *init_body* $>$): where *init_body* = (\circ ($\leftarrow_{i \in I}$ *user_channel* X_i) (\sqcup *cond_init* (\circ *cond_init* (*init_body*))), and *cond_init* = (\sqcup ($x_i[0] = \perp$) ($k_i = n_i = 0$)), and successive X_i , $i = 1, 2, \dots$ will be the following: $k\Omega_i[0]$ an initial meta-search procedure (default: as provided in this definition), $k_i, b_i, n_i, \Omega_i, A_i$ (defaults: $k_i = b_i = n_i = \infty, \Omega_i = A_i = \mathcal{A}$); simple and complex $\$$ -expressions definitions over $A_i \cup \Omega_i$ (default: no definitions);

initialize costs of simple $\$$ -expressions randomly and set reinforcement learning flag *reinf* = 1 (default: get costs of simple $\$$ -expressions from the user; *reinf* = 0); $\$i_1$ an aggregating cost function (default: addition), $\$i_2$ and $\$i_3$ search and solution specific cost functions (default: a standard cost function as defined in the next section);

Q_i optional goal of computation (default: $\min(\$i (k\Omega_i[t], x_i[t]))$);

$x_i[0]$ an initial $\$$ -expression solution (an initial state of LTS for the i -th agent) over alphabet $A_i \cup \Omega_i$. This resets gp_i flag to 0 (default: generate $x_i[0]$ randomly in the style of genetic programming and $gp_i = 1$);

/* receive from the user several values for initialization overwriting possibly the defaults. If atomic initialization fails re-initialize *init*. */

2. **Goal** ($:=$ (*goal_i*[t] ($k\Omega_i[t]$ $x_i[t]$)) $<$ *goal_body* $>$): where *goal_body* checks for the maximum predefined quantum of time (to avoid undecidability or too long verification) whether

goal state defined in the init phase has been reached. If the quantum of time expires, it returns false \perp .

3. Select Phase

(:= (sel $x_i[t]$) < (\sqcup cond_sel_exam (\circ cond_sel_exam sel_body)) >): where cond_sel_exam = (\sqcup ($k_i = 0$) ($b_i = 0$)) and sel_body builds the search tree with the branching factor b_i and depth k_i over alphabet $A_i \cup \Omega_i$ starting from the current state $x_i[t]$. For each state s derive actions a being multisets of simple $\$$ -expressions, and arriving in a new state s' . Actions and new states are found in two ways:

(a) if gp_i flag is set to 1 - by applying crossover/mutation (in the form of send and receive operating on LTS trees) to obtain a new state s' . A corresponding action between s and s' will be labeled as observationally congruent ε with neutral cost 0.

(b) if gp_i flag is set to 0 - by applying inference rules of LTS to a state.

Each simple $\$$ -expression in actions is labeled

- by its name if simple $\$$ -expression belongs to $A_i \cup \Omega_i$ and width and depth b_i, k_i are not exceeded,
- is renamed by strongly congruent ε with estimated cost if flag strongcong = 1 (default: renamed by weakly congruent ε with a neutral (zero) cost, strongcong = 0) if width b_i or depth k_i are exceeded /* hiding actions outside of the agent's width or depth search horizon, however not ignoring, but estimating their costs */.

For each new state s' check whether width/depth of the tree is exceeded, and whether it is a goal state. If so, s' becomes the leaf of the tree (for the current loop cycle), and no new actions are generated, otherwise continue to build the tree. If s' is a goal state, label it as a goal state.

4. Examine Phase

(:= (exam $x_i[t]$) < (\sqcup cond_sel_exam (\circ cond_sel_exam exam_body)) >): where exam_body prunes the search tree by selecting paths with minimal cost in cost choices and with maximal cost in adversary choices. Ties are broken randomly. In optimization, simple $\$$ -expressions belonging to $A_i - \Omega_i$ treat as observationally congruent ε with neutral cost (typically, equal to 0 like e.g., for a standard cost function) /* hiding agent's actions outside of its interests by ignoring their cost */.

5. Execute Phase

(:= (exec ($k\Omega_i[t]$ $x_i[t]$)) exec_body): where exec_body =

(\circ (\sqcup (\circ ($n_i = 0$)(goal_reached)(current_node = leaf_node_with_min_costs)) (\circ ($n_i = 0$)(goal_reached)(execute($x_i[t]$))(current_node = leaf_node))

(\circ ($n_i = 0$)(execute_n_i_steps($x_i[t]$))

(current_node = node_after_n_i_actions))) update_loop)

/* If $n_i = 0$ (offline search) and no goal state has been reached in the Select/Examine phase there will be no execution in this cycle. Pick up the most promising leaf node of the tree (with minimal cost) for expansion, i.e., make it a current node (root of the subtree expanded in the next cycle of the loop appended to an existing tree from the select phase, i.e., pruning will be invalidated to accommodate eventual corrections after cost updates). If $n_i = 0$ (offline search) and a goal state has been reached in the Select/Examine phase, execute optimal

$x_i[t]$ up to the leaf node using a tree constructed and pruned in the Select/Examine phase, or use LTS inference rules otherwise (for $gp = 1$). Make the leaf node a current node for a possible expansion (if it is not a goal state - it will be a root of a new tree) in the next cycle of the loop. If $n_i \neq 0$ (online search), execute optimal $x_i[t]$ up to at most n_i steps. Make the last state a current state - the root of the tree expanded in the next cycle of the loop. In execution simple $\$$ -expressions belonging to $\Omega_i - A_i$ will be executed by other agents.*/
The update_loop by default does nothing (executes silently ε with no cost) if update flag is reset. Otherwise if update = 1, then it gets from the user potentially all possible updates, e.g., new values of b_i, k_i, n_i and other parameters of $k\Omega[t]$, including costs of simple $\$$ -expressions, $\Omega_i, goal_i$. If update = 1 and the user does not provide own modifications (including possible overwriting the $k\Omega[t]$), then self-modification will be performed in the following way. If execution was interrupted (by receiving message from the user or environment invalidating solution found in the Select/Examine phase), then $n_i = 10$ if $n_i = \infty$, or $n_i = n_i - 1$ if $n_i \neq 0$, or $k_i = 10$ if $n_i = 0, k_i = \infty$, or $k_i = k_i - 1$ if $n_i = 0, k_i \neq \infty$. If execution was not interrupted increase $n_i = n_i + 1$ pending $0 < n_i \leq k_i$. If $n_i = k_i$ increase $k_i = k_i + 1, b_i = b_i + 1$. If cost of search ($\$_{2i}(k\Omega[t])$) larger than a predefined threshold decrease k_i and/or b_i , otherwise increase it. If reinforcement learning was set up $rein f = 1$ in the init phase, then cost of simple $\$$ -expressions will be modified by reinforcement learning.

The building of the LTS tree in the select phase for $gp_i = 0$ combines imperative and rule-based/logic styles of programming (we treat clause/production as a user-defined $\$$ -expression definition and call it by its name. This is similar to Robert Kowalski's dynamic interpretation of the left side of a clause as the name of procedure and the right side as the body of procedure.)

In the *init* and *exec/update* phase, in fact, a new search algorithm can be created (i.e., the old $k\Omega$ can be overwritten), and being completely different from the original $k\Omega$ -search. The original $k\Omega$ -search in self-modication changes the values of its control parameters mostly, i.e., k, n, b , but it could modify also *goal, sel, exam, exec* and $\$$.

Note that all parameters $k_i, n_i, \Omega_i, \$_i, A_i$, and \mathcal{A} can evolve in successive loop iterations. They are defined as the part of the *init* phase, and modified/updated at the end of the Execute phase *exec*. Note that they are associated with a specific choice of the meta-system: a $k\Omega$ -optimization search. For another meta-system, different control parameters are possible.

More details on the $k\Omega$ -search, including inference rules of the Labeled Transition System, observation and strong bisimulations and congruences, necessary and sufficient conditions to solve optimization, search optimization and total optimization problems, illustrating examples (including simulation by $k\Omega$ -optimization of A^* , Minimax, Traveling Salesman Problem, and other typical search algorithms), the details of implementations and applications, can be found in (Eberbach, 2005a, 2007).

4. The $\$$ -Calculus Expressiveness and its Support to Solve TM Undecidable Problems

To deal with undecidability, the $\$$ -calculus uses all three principles from introductory section: the infinity, interaction, and evolution principles:

- *infinity* - because of the infinity of the indexing set I in the $\$$ -calculus operators, it is clear that the $\$$ -calculus derives its expressiveness mostly from the *infinity* principle.
- *interaction* - if to assume that simple $\$$ -expressions may represent oracles, then the $\$$ -calculus can represent the interaction principle. Then we define an equivalent of the oracle as a user defined simple $\$$ -expression, that somehow in the manner of the “black-box” solves unsolvable problems (however, we do not know how).
- *evolution* - the $k\Omega$ -optimization may be evolved to a new (and hopefully) more powerful problem solving method

It is easier and “cleaner” to think about implementation of unbounded (infinitary) concepts, than about implementation of oracles. The implementation of scalable computers (e.g., scalable massively parallel computers or unbounded growth of Internet) allows to think about a reasonable approximation of the implementation of *infinity* (and, in particular, the π -calculus, or the $\$$ -calculus). At this point, it is not clear how to implement oracles (as Turing stated *an oracle cannot be a machine*, i.e., implementable by mechanical means), and as the result, the models based on them. One of potential implementation of oracles could be an infinite lookup table with stored all results of the decision for TM. The quite different story is how to initialize such infinite lookup table and how to search it effectively using for instance hash indices or B+ trees.

The expressiveness of the $\$$ -calculus is not worse than the expressiveness of Turing Machines, i.e., it is straightforward to show how to encode λ -calculus (Church, 1936, 1941) in $\$$ -calculus (Eberbach, 2006, 2007). In (Eberbach, 2005a) it has been demonstrated, how some other models of computation, more expressive than Turing Machines, can be simulated in the $\$$ -calculus. This includes the π -calculus, Interaction Machines, cellular automata, neural networks, and random automata networks.

It is interesting that the $\$$ -calculus can solve in the limit the halting problem of the Universal Turing Machine, and approximate the solution of the halting/optimization problem of the $\$$ -calculus. This is a very interesting result, because, if correct, besides the $\$$ -calculus, it may suggest that a self-modifying program using infinitary means may approximate the solution of its own decision (halting or optimization) problem.

4.1. Solving the Turing Machine Halting Problem and Approximating the Universal Search Algorithm

The results from Eberbach (2006, 2007) justify that the $\$$ -calculus is more expressive than the TM, and may represent non-algorithmic computation. In Eberbach (2006, 2007) three ways how the $\$$ -calculus solves the halting problem of the Universal Turing Machine using either an *infinity*, *evolution*, or *interaction principle*.

Theorem 4.1 (On solution of the halting problem of UTM by $\$$ -calculus (Eberbach, 2006, 2007)). *The halting problem for the Universal Turing Machine is solvable by the $\$$ -calculus.*

Proof. (Outline): In the infinity principle $\$$ -calculus taking an instance of TM code and its input runs an infinite number of steps of TM (same idea like for example in Infinite Time TM). Of course, in infinity the answer for halting will be yes or no. It is a matter of philosophical discussion whether

getting a definitive answer in infinity constitutes an answer at all. In the interaction principle, $\$$ -calculus $k\Omega$ -search gets an answer from oracle, and in the evolution principle, the TM is evolved to TM with oracle. Assuming that a TM with an oracle can be encoded as a nonrecursive function ($\$$ -expression) using a binary alphabet and an ordinary TM can be (of course) encoded as a binary string, thus a simple binary mutation changing one binary input to another can convert by chance an ordinary Turing Machine to the Turing Machine with an oracle (Turing, 1939). The probability of such event is very small, and because nobody has implemented the Turing Machine with an oracle (although we know what its transition function may look like - see, e.g. (Kozen, 1997)). We may have the big problem with the recognition of this encoding, thus even if theoretically possible, so far nobody has been able to detect the potential existence of the Turing Machine with an oracle. \square

We know from the theory of computation that the search for the universal algorithm is a futile effort. If it was not so, such algorithm could be used immediately to solve the halting problem of TM.

We will show how the $\$$ -calculus can help potentially for the solution of the best search algorithm by approximating it. The best search algorithm will be in the sense of the optimum: finding the best-quality solutions/search algorithms for the all possible problems. The trouble and undecidability is caused by the requirement to cover exactly *all* possible problems. The number of possible algorithms is enumerable, however, the number of possible problems is infinite, but not enumerable (problems/languages are all possible subsets of all algorithms), see, e.g., (Hopcroft *et al.*, 2001).

Theorem 4.2 (On approximating the universal search algorithm (Eberbach, 2006, 2007)). *The $k\Omega$ -optimization taking itself as its input will converge with an arbitrarily small error in finite time to the universal search algorithm if search is complete and elitist selection strategy is used.*

Proof. (Outline): In the above approach completeness guarantees that no solution will be missed and elitist selection allows to preserve the best solution found so far. In other words, the $k\Omega$ -search taking as its input itself, produces in the finite time better versions of itself and in infinity reaches the optimum (pending that it exists). This allows to approximate the best search algorithm in the finite time. In particular, this can be used for approximated solution of the UTM halting problem. \square

In such way progress in mathematics or computer science (both being undecidable) is done. Proving all theorems (existing and not derived yet) in mathematics or computer science is impossible (the Entscheidungsproblem - Hilbert's decision problem is undecidable (Whitehead & Russell, 1910, 1912, 1913; Turing, 1937; Hopcroft *et al.*, 2001)). However, in spite of that, new generations of mathematicians and computer scientists work and generate new theorems and improve indefinitely the current state of art of mathematics and computer science. The famous unsolvable Entscheidungsproblem does not prevent scientists from discovering new theorems, improving our knowledge of mathematics, but we will never be able to write all theorems (unless to wait for eternity).

4.2. Deciding the Diagonalization Language, Nontrivial Properties, Solving Post Correspondence Problem and Busy Beaver Problem

In (Eberbach, 2003) several open problems have been posed. We will present solutions to some of them.

The diagonalization language L_d is an example of the language that is believed even tougher than L_u , i.e., language of UTM accepting words w for arbitrary TM M . L_d is non-RE, i.e., it does not exist any TM accepting it.

Definition 4.1. The diagonalization language L_d consists of all strings w such that TM M whose code is w does not accept when given w as input.

The existence of diagonalization language that cannot be accepted by any TM is proven by diagonalization table with “dummy”, i.e., not real/true values. Of course, there are many diagonalization language encodings possible that depend how transitions of TMs are encoded. This means that there are infinitely many different L_d language instances (but nobody wrote a specific example of L_d). Solving the halting problem of UTM, can be used for a “constructive” proof of the diagonalization language decidability demonstrating all strings belonging to the language.

Theorem 4.3 (Deciding asymptotically the diagonalization language). *The diagonalization language L_d is $\$$ -calculus asymptotically decidable pending that language of halting UTM L_u is decidable.*

Proof. By Theorem 4.1 we fill a characteristic vector for each TM in diagonalization table, i.e., for each TM M_i and for each its input string w_j , $i, j = 0, 1, 2, \dots$, we write 1 if w_j is accepted by M_i and 0 otherwise. This is always possible pending that halting of UTM is solvable. For each M_i we look at w_i (the diagonal value) and flip its bit to opposite value. Now for each w_i , $i = 0, 1, 2, \dots$, we construct a characteristic vector for L_d . For each accepted string w_i we construct a finite automaton FA_i accepting exactly one word w_i (always trivially possible in finite time). We construct an infinite parallel composition - $\$$ -calculus $\$$ -expression $(\parallel_i FA_i)$ and put to it an arbitrary input string w . If $(\parallel_i FA_i)$ accepts w (i.e., one of FA accepts) then L_d accepts. If no FA accepts then L_d does not accept either. \square

In such a way we can decide in $\$$ -calculus a language L_d that is not possible to decide in the TM model.

In an analogous way we can decide other TM unsolvable languages/problems.

The language L_{ne} consisting of all binary encoded TMs whose language is not empty, i.e., $L_{ne} = \{M \mid L(M) \neq \emptyset\}$ is known to be recursively enumerable but not recursive, and its complement language $L_e = \{M \mid L(M) = \emptyset\}$ consisting of all binary encoded TMs whose language is empty is known to be non recursively enumerable.

Theorem 4.4 (Deciding asymptotically L_{ne} and L_e). *The languages L_{ne} and L_e are $\$$ -calculus asymptotically decidable pending that language of halting UTM L_u is decidable.*

Proof. It is enough to look at the diagonalization table found after solving UTM L_u halting problem. The rows containing at least one 1 allow to decide L_{ne} , and complementary rows consisting of all 0s allow to decide L_e . \square

We can solve nontrivial properties (nonempty proper subsets of all RE languages), i.e., to prove solvability of Rice theorem (Rice, 1953) using hypercomputation.

Theorem 4.5 (Deciding asymptotically nontrivial properties). *Every nontrivial property is \mathcal{H} -calculus asymptotically decidable pending that language of halting UTM L_u is decidable.*

Proof. We identify the proper subset of the rows from the diagonalization table satisfying a specific nonempty property. It is necessary to translate specific property in terms of corresponding 1s and 0s in characteristic vectors for each TM M_i (e.g., to translate which rows correspond to the empty language, a finite language, a regular language, context-free language, context-sensitive language). \square

It is also possible to decide PCP and Busy Beaver Problem.

Definition 4.2. The TM undecidable Post Correspondence Problem (PCP) (Post, 1946) asks, given two lists of the same number of strings over the same alphabet, whether we can pick a sequence of corresponding strings from the two lists and form the same string by concatenation.

Theorem 4.6 (Deciding asymptotically PCP). *The Post Correspondence Problem is asymptotically decidable pending that language of halting UTM L_u is decidable.*

Proof. As the halting or terminal state of the TM solving PCP we put the condition whether both lists produce the same string. \square

Definition 4.3. The TM undecidable Busy Beaver Problem (BBP) (Rado, 1962) considers a deterministic 1-tape Turing machine with unary alphabet $\{1\}$ and tape alphabet $\{1, B\}$, where B represents the tape blank symbol. TM starts with an initial empty tape and accepts by halting. For the arbitrary number of states $n = 0, 1, 2, \dots$ TM tries to compute two functions: the maximum number of 1s written on tape before halting (known as the busy beaver function $\Sigma(n)$) and the maximum number of steps before halting (known as the maximum shift function $S(n)$).

Theorem 4.7 (Deciding asymptotically BBP). *The Busy Beaver Problem is asymptotically decidable pending that language of halting UTM L_u is decidable.*

Proof. As the halting state of the TM solving BBP we put the condition whether $\Sigma(n)$ and $S(n)$ have been computed. \square

5. Conclusions

In the paper some hypercomputation solutions of Turing Machine unsolvable problems have been presented. We demonstrate that the solution of the halting problem is like to solve polynomially one NP-complete problem to resolve famous dilemma $\mathcal{P} ? = \mathcal{NP}$, i.e., it breaks the whole hierarchical puzzle of unsolvability. Namely, solving the halting problem of UTM is pivotal to solve many other Turing Machine unsolvable problems, including to decide the diagonalization language, nontrivial properties, PCP and BBP.

However, hypercomputational models are still not well researched and many scientists vigorously oppose the idea that computations going beyond Turing Machine are possible at all. Very

little is known about the hierarchy (or at least relations) between hypercomputational models. We do not know enough about the implementability of hypercomputation at least at the level similar to quantum computing or biomolecular computing implementations. We do not know the limits of computability in hypercomputational models. We do not know whether such limits exist at all. In other words, hypercomputation is in the situation similar like Turing, Church and Gödel were in 1930s before the whole digital computers explosive growth started. Whether these hopes and fears about hypercomputation will be materialized is a quite different story.

References

- Aho, Alfred V. (2011). Ubiquity symposium: Computation and computational thinking. *Ubiquity*.
- Burgin, M. S. (2004). *Super-Recursive Algorithms (Monographs in Computer Science)*. SpringerVerlag.
- Church, A. (1936). An unsolvable problem of elementary number theory. *American Journal of Mathematics* **58**(2), 345–363.
- Church, A. (1941). *The Calculi of Lambda Conversion*. Princeton, N.J., Princeton University Press. New York, NY, USA.
- Cooper, B. (2012). Turing’s titanic machine?. *Commun. ACM* **55**(3), 74–83.
- Eberbach, E. (1997). A generic tool for distributed AI with matching as message passing. In: *Tools with Artificial Intelligence, 1997. Proceedings., Ninth IEEE International Conference on*. pp. 11–18.
- Eberbach, E. (2003). Is entscheidungsproblem solvable? beyond undecidability of Turing machines and its consequence for computer science and mathematics. In: (ed. J. C. Misra) *Computational Mathematics, Modelling and Algorithms*. pp. 1–32. Narosa Publishing House, New Delhi.
- Eberbach, E. (2005a). $\$$ -Calculus of bounded rational agents: Flexible optimization as search under bounded resources in interactive systems. *Fundam. Inf.* **68**(1-2), 47–102.
- Eberbach, E. (2005b). Toward a theory of evolutionary computation. *Biosystems* **82**(1), 1 – 19.
- Eberbach, E. (2006). Expressiveness of the π -Calculus and the $\$$ -Calculus. In: *Proc. 2006 World Congress in Comp. Sci., Comp. Eng., & Applied Computing, The 2006 Intern. Conf. on Foundations of Computer Science FCS’06, Las Vegas, Nevada*. pp. 24–30.
- Eberbach, E. (2007). The $\$$ -calculus process algebra for problem solving: A paradigmatic shift in handling hard computational problems. *Theoretical Computer Science* **383**(23), 200 – 243. Complexity of Algorithms and Computations.
- Eberbach, E. and M. Burgin (2009). On foundations of evolutionary computation: An evolutionary automata approach. In: (ed. Hongwei Mo): *Handbook of Research on Artificial Immune Systems and Natural Computing: Applying Complex Adaptive Technologies, Section II: Natural Computing, Section II.1: Evolutionary Computing, Chapter XVI, Medical Information Science Reference/IGI Global, Hershey*. pp. 342–360. New York.
- Eberbach, E., D. Goldin and P. Wegner (2004). Turings ideas and models of computation. In: *Alan Turing: Life and Legacy of a Great Thinker* (Christof Teuscher, Ed.). pp. 159–194. Springer Berlin Heidelberg.
- Eberbach, Eugene and Peter Wegner (2003). Beyond Turing machines. *Bulletin of the EATCS* pp. 279–304.
- Hopcroft, J. E., R. Motwani and J. D. Ullman (2001). Introduction to automata theory, languages, and computation, 2nd edition. *SIGACT News* **32**(1), 60–65.
- Horvitz, E. and S. Zilberstein (2001). Computational tradeoffs under bounded resources. *Artificial Intelligence* **126**(12), 1 – 4. Tradeoffs under Bounded Resources.
- Kozen, D. C. (1997). *Automata and Computability*. Springer-Verlag.
- Milner, R. (1999). *Communicating and Mobile Systems: The π -calculus*. Cambridge University Press. New York, NY, USA.

- Milner, R., J. Parrow and D. Walker (1992). A calculus of mobile processes, i. *Information and Computation* **100**(1), 1–40.
- Post, E. (1946). A variant of a recursively unsolvable problem. *Bull. Amer. Math. Soc.* **52**(4), 264–268.
- Rado, T. (1962). On non-computable functions. *Bell System Technical Journal* **41**(3), 877–884.
- Rice, H. G. (1953). Classes of recursively enumerable sets and their decision problems. *Trans. Amer. Math. Soc.* **74**, 358–366.
- Russell, S. and P. Norvig (2002). *Artificial Intelligence: A Modern Approach (2nd Edition)*. Prentice Hall.
- Syropoulos, A. (2007). *Hypercomputation: Computing Beyond the Church-Turing Barrier (Monographs in Computer Science)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.
- Turing, A. M. (1937). On computable numbers, with an application to the entscheidungsproblem. *Proceedings of the London Mathematical Society* **s2-42**(1), 230–265.
- Turing, A. M. (1939). Systems of logic based on ordinals. *Proceedings of the London Mathematical Society* **s2-45**(1), 161–228.
- Wegner, P., E. Eberbach and M. Burgin (2012). Computational completeness of interaction machines and Turing machines. In: *Turing-100* (Andrei Voronkov, Ed.). Vol. 10 of *EPiC Series*. EasyChair. pp. 405–414.
- Whitehead, A. N. and B. Russell (1910, 1912, 1913). *Principia mathematica, vol.1, 1910, vol.2, 1912, vol.3, 1913*. Cambridge Univ. Press. Cambridge.



Coupled Systems of Fractional Integro-Differential Equations Involving Several Functions

Zoubir Dahmani^{a,*}, Mohamed Amin Abdellaoui^a, Mohamed Houas^b

^aLaboratory LPAM, Faculty SEI, UMAB University, Algeria

^bLaboratory FIMA, Faculty ST, University of Khemis Miliana, Algeria

Abstract

This paper studies the existence of solutions for a coupled system of nonlinear fractional integro-differential equations involving Riemann-Liouville integrals with several continuous functions. New existence and uniqueness results are established using Banach fixed point theorem, and other existence results are obtained using Schaefer fixed point theorem. Some illustrative examples are also presented.

Keywords: Caputo derivative, fixed point, integro-differential system, existence, uniqueness, Riemann-Liouville integral.

2010 MSC: 34A34, 34B10.

1. Introduction

The differential equations of fractional order arise in many scientific disciplines, such as physics, chemistry, control theory, signal processing and biophysics. For more details, we refer the reader to (Kilbas & Marzan, 2005; Lakshmikantham & Vatsala, 2008; Su, 2009) and the references therein. Recently, there has been a significant progress in the investigation of these equations, (see (Anber *et al.*, 2013; Bengrine & Dahmani, 2012; Cui *et al.*, 2012; Wang *et al.*, 2010; Zhang, 2006)). On the other hand, the study of coupled systems of fractional differential equations is also of a great importance. Such systems occur in various problems of applied science. For some recent results on the fractional systems, we refer the reader to (Abdellaoui *et al.*, 2013; Bai & Fang, 2004; Gaber & Brikaa, 2012; Gafiychuk *et al.*, 2008).

*Corresponding author

Email addresses: zzdahmani@yahoo.fr (Zoubir Dahmani), abdellaouiamine13@yahoo.fr (Mohamed Amin Abdellaoui), houasmed@yahoo.fr (Mohamed Houas)

In this paper, we discuss the existence and uniqueness of solutions for the following coupled system of fractional integro-differential equations:

$$\begin{cases} D^\alpha u(t) = f_1(t, u(t), v(t)) + \sum_{i=1}^m \int_0^t \frac{(t-s)^{\alpha_i-1}}{\Gamma(\alpha_i)} \varphi_i(s) g_i(s, u(s), v(s)) ds, \\ D^\beta v(t) = f_2(t, u(t), v(t)) + \sum_{i=1}^m \int_0^t \frac{(t-s)^{\beta_i-1}}{\Gamma(\beta_i)} \phi_i(s) h_i(s, u(s), v(s)) ds, \\ u(0) = a > 0, v(0) = b > 0, t \in [0, 1] \end{cases} \quad (1.1)$$

where D^α, D^β denote the Caputo fractional derivatives, $0 < \alpha < 1, 0 < \beta < 1, \alpha_i, \beta_i$ are non negative real numbers, φ_i and ϕ_i are continuous functions, $m \in \mathbb{N}^*$, f_1, f_2 and g_i and $h_i, i = 1, \dots, m$, are functions that will be specified later.

The paper is organized as follows: In section 2, we present some preliminaries and lemmas. Section 3 is devoted to existence of solutions of problem (1.1). In the last section, some examples are presented to illustrate our results.

2. Preliminaries

The following notations, definitions and lemmas will be used throughout this paper.

Definition 2.1. The Riemann-Liouville fractional integral operator of order $\alpha > 0$, for $f \in L^1([a, b], \mathbb{R})$ is defined by:

$$I^\alpha f(t) = \int_a^t \frac{(t-\tau)^{\alpha-1}}{\Gamma(\alpha)} f(\tau) d\tau, \quad a \leq t \leq b, \quad (2.1)$$

where $\Gamma(\alpha) := \int_0^\infty e^{-u} u^{\alpha-1} du$.

Definition 2.2. The fractional derivative of $f \in C^n([a, b], \mathbb{R}), n \in \mathbb{N}^*$, in the sense of Caputo, of order $\alpha, n-1 < \alpha < n$ is defined by:

$$D^\alpha f(t) = \int_a^t \frac{(t-\tau)^{n-\alpha-1}}{\Gamma(n-\alpha)} f^{(n)}(\tau) d\tau, \quad t \in [a, b]. \quad (2.2)$$

For more details about fractional calculus, we refer the reader to (Mainardi, 1997). The following lemmas give some properties of Riemann-Liouville integrals and Caputo fractional derivatives (Kilbas & Marzan, 2005; Lakshmikantham & Vatsala, 2008):

Lemma 2.1. Given $f \in L^1([a, b], \mathbb{R})$, then for all $t \in [a, b]$ we have $I^r I^s f(t) = I^{r+s} f(t)$, for $r, s > 0$. $D^s I^s f(t) = f(t)$, for $s > 0$. $D^r I^s f(t) = I^{s-r} f(t)$, for $s > r > 0$.

To study the coupled system (1.1), we need the following two lemmas (Kilbas & Marzan, 2005):

Lemma 2.2. For $n - 1 < \alpha < n$, where $n \in \mathbb{N}^*$, the general solution of the equation $D^\alpha x(t) = 0$ is given by

$$x(t) = c_0 + c_1 t + c_2 t^2 + \dots + c_{n-1} t^{n-1}, \tag{2.3}$$

where $c_i \in \mathbb{R}, i = 0, 1, 2, \dots, n - 1$.

Lemma 2.3. Let $n - 1 < \alpha < n$, where $n \in \mathbb{N}^*$. Then, for $x \in C^n([0, 1], \mathbb{R})$, we have

$$I^\alpha D^\alpha x(t) = x(t) + c_0 + c_1 t + c_2 t^2 + \dots + c_{n-1} t^{n-1}, \tag{2.4}$$

for some $c_i \in \mathbb{R}, i = 0, 1, 2, \dots, n - 1, n = [\alpha] + 1$.

We prove the following auxiliary lemma:

Lemma 2.4. Let $f, R_i, K_i \in C([0, 1], \mathbb{R}), i = 1, \dots, m$. The solution of the problem

$$D^\alpha x(t) = f(t) + \sum_{i=1}^m \int_0^t \frac{(t-s)^{\alpha_i-1}}{\Gamma(\alpha_i)} R_i(s) K_i(s) ds, 0 < \alpha < 1, \quad \alpha_i > 0 \tag{2.5}$$

with the condition, $x(0) = x_0^* \in \mathbb{R}_+^*$, is given by

$$x(t) = \int_0^t \frac{(t-s)^{\alpha-1}}{\Gamma(\alpha)} f(s) ds + \sum_{i=1}^m \int_0^t \frac{(t-s)^{\alpha+\alpha_i-1}}{\Gamma(\alpha+\alpha_i)} R_i(s) K_i(s) ds + x_0^*. \tag{2.6}$$

Proof. Setting

$$y(t) = x(t) - I^\alpha f(t) - \sum_{i=1}^m I^{\alpha+\alpha_i} R_i(t) K_i(t). \tag{2.7}$$

Thanks to the linearity of D^α , we get

$$D^\alpha y(t) = D^\alpha x(t) - D^\alpha I^\alpha f(t) - \sum_{i=1}^m D^\alpha I^{\alpha+\alpha_i} R_i(t) K_i(t). \tag{2.8}$$

By lemma 2.2, yields

$$D^\alpha y(t) = D^\alpha x(t) - f(t) - \sum_{i=1}^m I^{\alpha_i} R_i(t) K_i(t). \tag{2.9}$$

Thus, (2.5) is equivalent to $D^\alpha y(t) = 0$.

Finally, thanks to lemma 2.3, we obtain that $y(t)$ is constant, i.e., $y(t) = y(0) = x(0) = x_0^*$, and the proof of lemma 2.4 is achieved. \square

3. Main Results

We introduce in this paragraph the following assumptions:

(H1) : There exist non negative real numbers $\mu_j, \nu_j; j = 1, 2$ and $l_i, m_i, n_i, k_i, i = 1, \dots, m$, such that for all $t \in [0, 1]$ and $(u_1, v_1), (u_2, v_2) \in \mathbb{R}^2$, we have

$$|f_j(t, u_2, v_2) - f_j(t, u_1, v_1)| \leq \mu_j |u_2 - u_1| + \nu_j |v_2 - v_1|, j = 1, 2$$

$$|g_i(t, u_2, v_2) - g_i(t, u_1, v_1)| \leq l_i |u_2 - u_1| + m_i |v_2 - v_1|, i = 1, \dots, m,$$

and,

$$|h_i(t, u_2, v_2) - h_i(t, u_1, v_1)| \leq n_i |u_2 - u_1| + k_i |v_2 - v_1|, i = 1, \dots, m$$

with

$$\bar{L} = \max(\mu_1, \mu_2, \nu_1, \nu_2, l_i, m_i, n_i, k_i, i = 1, \dots, m.)$$

(H2) : The functions f_1, f_2, g_i and $h_i : [0, 1] \times \mathbb{R}^2 \rightarrow \mathbb{R}$ are continuous for each $i = 1, \dots, m$.

(H3) : There exist positive real numbers $L_1, L_2, L'_i, L''_i, i = 1, \dots, m$, such that

$$|f_1(t, u, v)| \leq L_1, |g_i(t, u, v)| \leq L'_i, |f_2(t, u, v)| \leq L_2,$$

$$|h_i(t, u, v)| \leq L''_i, t \in [0, 1], (u, v) \in \mathbb{R}^2.$$

Our first result is given by:

Theorem 3.1. Assume that (H1) holds and setting

$$M_1 : = \frac{1}{\Gamma(\alpha + 1)} + \sum_{i=1}^m \frac{\|\varphi_i\|_\infty}{\Gamma(\alpha + \alpha_i + 1)},$$

$$M_2 : = \frac{1}{\Gamma(\beta + 1)} + \sum_{i=1}^m \frac{\|\phi_i\|_\infty}{\Gamma(\beta + \beta_i + 1)}.$$

If

$$\bar{L}(M_1 + M_2) < 1, \quad (3.1)$$

then, the system (1.1) has exactly one solution on $[0, 1]$.

Proof. Setting $X := C([0, 1], \mathbb{R})$. This space, equipped with the norm $\|\cdot\|_X = \|\cdot\|_\infty$ defined by $\|f\|_\infty = \sup\{|f(x)|, x \in [0, 1]\}$, is a Banach space. Also, the product space $(X \times X, \|(u, v)\|_{X \times X})$ is a Banach space, with $\|(u, v)\|_{X \times X} = \|u\|_X + \|v\|_X$.

Consider now the operator $\Psi : X \times X \rightarrow X \times X$ defined by

$$\Psi(u, v)(t) = \left(\Psi_1(u, v)(t), \Psi_2(u, v)(t) \right), \quad (3.2)$$

where

$$\Psi_1(u, v)(t) = \int_0^t \frac{(t-s)^{\alpha-1}}{\Gamma(\alpha)} f_1(s, u(s), v(s)) ds + \sum_{i=1}^m \int_0^t \frac{(t-s)^{\alpha+\alpha_i-1}}{\Gamma(\alpha + \alpha_i)} \varphi_i(s) g_i(s, u(s), v(s)) ds + a. \quad (3.3)$$

and

$$\Psi_2(u, v)(t) = \int_0^t \frac{(t-s)^{\beta-1}}{\Gamma(\beta)} f_2(s, u(s), v(s)) ds + \sum_{i=1}^m \int_0^t \frac{(t-s)^{\beta+\beta_i-1}}{\Gamma(\beta+\beta_i)} \varphi_i(s) g_i(s, u(s), v(s)) ds + b. \tag{3.4}$$

We shall show that Ψ is contractive: Let $(u_1, v_1), (u_2, v_2) \in X \times X$. Then, for each $t \in [0, 1]$, we have

$$\begin{aligned} |\Psi_1(u_2, v_2)(t) - \Psi_1(u_1, v_1)(t)| \leq & \int_0^t \frac{(t-s)^{\alpha-1}}{\Gamma(\alpha)} ds \times \sup_{0 \leq s \leq 1} |f_1(s, u_2(s), v_2(s)) - f_1(s, u_1(s), v_1(s))| \\ & + \sum_{i=1}^m (\sup_{0 \leq s \leq 1} |\varphi_i(s)| \int_0^t \frac{(t-s)^{\alpha+\alpha_i-1}}{\Gamma(\alpha+\alpha_i)} ds \\ & \times \sup_{0 \leq s \leq 1} |g_i(s, u_2(s), v_2(s)) - g_i(s, u_1(s), v_1(s))|. \end{aligned} \tag{3.5}$$

Therefore,

$$\begin{aligned} |\Psi_1(u_2, v_2)(t) - \Psi_1(u_1, v_1)(t)| \leq & \frac{1}{\Gamma(\alpha+1)} \sup_{0 \leq s \leq 1} |f_1(s, u_2(s), v_2(s)) - f_1(s, u_1(s), v_1(s))| \\ & + \sum_{i=1}^m \frac{\|\varphi_i\|_\infty}{\Gamma(\alpha+\alpha_i+1)} \sup_{0 \leq s \leq 1} |g_i(s, u_2(s), v_2(s)) - g_i(s, u_1(s), v_1(s))|. \end{aligned} \tag{3.6}$$

Using (H1), we can write:

$$\begin{aligned} |\Psi_1(u_2, v_2)(t) - \Psi_1(u_1, v_1)(t)| \leq & \frac{\bar{L}}{\Gamma(\alpha+1)} \left(\sup_{0 \leq t \leq 1} |u_2(t) - u_1(t)| + \sup_{0 \leq t \leq 1} |v_2(t) - v_1(t)| \right) \\ & + \sum_{i=1}^m \frac{\|\varphi_i\|_\infty \bar{L}}{\Gamma(\alpha+\alpha_i+1)} \left(\sup_{0 \leq t \leq 1} |u_2(t) - u_1(t)| + \sup_{0 \leq t \leq 1} |v_2(t) - v_1(t)| \right). \end{aligned} \tag{3.7}$$

This implies that

$$|\Psi_1(u_2, v_2)(t) - \Psi_1(u_1, v_1)(t)| \leq M_1 \bar{L} (\|u_2 - u_1\|_X + \|v_2 - v_1\|_X). \tag{3.8}$$

And consequently,

$$\|\Psi_1(u_2, v_2) - \Psi_1(u_1, v_1)\|_X \leq M_1 \bar{L} \|(u_2 - u_1, v_2 - v_1)\|_{X \times X}. \tag{3.9}$$

With the same arguments as before, we can write

$$\|\Psi_2(u_2, v_2) - \Psi_2(u_1, v_1)\|_X \leq M_2 \bar{L} \|(u_2 - u_1, v_2 - v_1)\|_{X \times X}. \quad (3.10)$$

Finally, using (3.9) and (3.10), we deduce that

$$\|\Psi(u_2, v_2) - \Psi(u_1, v_1)\|_{X \times X} \leq \bar{L}(M_1 + M_2) \|(u_2 - u_1, v_2 - v_1)\|_{X \times X}. \quad (3.11)$$

Thanks to (3.1), we conclude that Ψ is a contraction mapping. Hence, by Banach fixed point theorem, there exists a unique fixed point which is a solution of (1.1). \square

The second result is the following:

Theorem 3.2. Assume that (H2) and (H3) are satisfied and $L'_i \leq L_1, L''_i \leq L_2, i = 1, \dots, m$. Then problem (1.1) has at least one solution on $[0, 1]$.

Proof. First of all, we show that the operator T is completely continuous.

Step 1: Let us take $\gamma > 0$ and $B_\gamma := \{(u, v) \in X \times X; \|(u, v)\|_{X \times X} \leq \gamma\}$. Now, assume that (H3) holds, and $L'_i \leq L_1, L''_i \leq L_2$. Then for $(u, v) \in B_\gamma$, we have

$$\begin{aligned} |\Psi_1(u, v)(t)| &\leq a + \frac{t^\alpha}{\Gamma(\alpha+1)} \sup_{0 \leq t \leq 1} |f_1(t, u(t), v(t))| \\ &+ \sum_{i=1}^m \frac{\|\varphi_i\|_\infty t^{\alpha+\alpha_i}}{\Gamma(\alpha+\alpha_i+1)} \sup_{0 \leq t \leq 1} |g_i(t, u(t), v(t))|, t \in [0, 1]. \end{aligned} \quad (3.12)$$

Hence, we obtain

$$\|\Psi_1(u, v)\|_X \leq L_1 M_1 + a < +\infty. \quad (3.13)$$

With the same arguments, we have

$$\|\Psi_2(u, v)\|_X \leq L_2 M_2 + b < +\infty. \quad (3.14)$$

Then, by (23) and (24), we can state that $\|T(u, v)\|_{X \times X}$ is bounded by C , where

$$C := L_1 M_1 + L_2 M_2 + a + b. \quad (3.15)$$

Step 2: Let $t_1, t_2 \in [0, 1], t_1 < t_2$ and $(u, v) \in B_\gamma$. We have

$$\begin{aligned} |\Psi_1(u, v)(t_2) - \Psi_1(u, v)(t_1)| &\leq \left| \int_0^{t_2} \frac{(t_2-s)^{\alpha-1}}{\Gamma(\alpha)} f_1(s, u(s), v(s)) ds - \int_0^{t_1} \frac{(t_1-s)^{\alpha-1}}{\Gamma(\alpha)} f_1(s, u(s), v(s)) ds \right| \\ &+ \left| \sum_{i=1}^m \int_0^{t_2} \frac{(t_2-s)^{\alpha+\alpha_i-1}}{\Gamma(\alpha+\alpha_i)} \varphi_i(s) g_i(s, u(s), v(s)) ds - \sum_{i=1}^m \int_0^{t_1} \frac{(t_1-s)^{\alpha+\alpha_i-1}}{\Gamma(\alpha+\alpha_i)} \varphi_i(s) g_i(s, u(s), v(s)) ds \right|. \end{aligned} \quad (3.16)$$

Thus, we get

$$|\Psi_1(u, v)(t_2) - \Psi_1(u, v)(t_1)| \leq \frac{L_1(t_2^\alpha - t_1^\alpha + (t_2 - t_1)^\alpha)}{\Gamma(\alpha + 1)} + \sum_{i=1}^m \frac{L_1 \|\varphi_i\|_\infty (t_2^{\alpha+\alpha_i} - t_1^{\alpha+\alpha_i} + (t_2 - t_1)^{\alpha+\alpha_i})}{\Gamma(\alpha + \alpha_i + 1)}. \quad (3.17)$$

Analogously, we can obtain

$$|\Psi_2(u, v)(t_2) - \Psi_2(u, v)(t_1)| \leq \frac{L_2(t_2^\beta - t_1^\beta + (t_2 - t_1)^\beta)}{\Gamma(\beta + 1)} + \sum_{i=1}^m \frac{L_2 \|\varphi_i\|_\infty (t_2^{\beta+\beta_i} - t_1^{\beta+\beta_i} + (t_2 - t_1)^{\beta+\beta_i})}{\Gamma(\beta + \beta_i + 1)}. \tag{3.18}$$

Therefore,

$$|\Psi(u, v)(t_2) - \Psi(u, v)(t_1)| \leq \frac{L_1(t_2^\alpha - t_1^\alpha + (t_2 - t_1)^\alpha)}{\Gamma(\alpha + 1)} + \sum_{i=1}^m \frac{L_1 \|\varphi_i\|_\infty (t_2^{\alpha+\alpha_i} - t_1^{\alpha+\alpha_i} + (t_2 - t_1)^{\alpha+\alpha_i})}{\Gamma(\alpha + \alpha_i + 1)} + \frac{L_2(t_2^\beta - t_1^\beta + (t_2 - t_1)^\beta)}{\Gamma(\beta + 1)} + \sum_{i=1}^m \frac{L_2 \|\varphi_i\|_\infty (t_2^{\beta+\beta_i} - t_1^{\beta+\beta_i} + (t_2 - t_1)^{\beta+\beta_i})}{\Gamma(\beta + \beta_i + 1)}. \tag{3.19}$$

As $t_2 \rightarrow t_1$, the right-hand side of (3.19) tends to zero. Then, as a consequence of Steps 1, 2, and by Arzela-Ascoli theorem, we conclude that Ψ is completely continuous.

Next, we consider the set:

$$\Omega = \{(u, v) \in X \times X; (u, v) = \lambda T(u, v), 0 < \lambda < 1\}. \tag{3.20}$$

We shall show that Ω is bounded:

Let $(u, v) \in \Omega$, then $(u, v) = \lambda \Psi(u, v)$, for some $0 < \lambda < 1$. Hence, for $t \in [0, 1]$, we have:

$$u(t) = \lambda \Psi_1(u, v)(t), v(t) = \lambda \Psi_2(u, v)(t). \tag{3.21}$$

Thus,

$$\|(u, v)\|_{X \times X} = \lambda \|\Psi(u, v)\|_{X \times X}. \tag{3.22}$$

Thanks to (H_3) ,

$$\|(u, v)\|_{X \times X} \leq \lambda C, \tag{3.23}$$

where C is defined by (3.15). Therefore, Ω is bounded.

As a conclusion of Schaefer fixed point theorem, we deduce that Ψ has at least one fixed point, which is a solution of (1.1). □

4. Examples

Example 4.1. Consider the following fractional system:

$$\begin{cases} D^{\frac{1}{2}} u(t) = \left(\frac{\sin(u(t)+v(t))}{18(\ln(t+1)+1)} + 6 \right) + \int_0^t \frac{(t-s)^{\frac{1}{2}}}{\Gamma(\frac{3}{2})} \left(\frac{\exp(-s)}{18(s+1)} \frac{\sin(u(s)+v(s))}{18(s+5)} \right) ds, t \in [0, 1], \\ D^{\frac{1}{2}} v(t) = \frac{\sin u(s)+\sin v(s)}{16(t \exp(t^2)+1)} + \int_0^t \frac{(t-s)^{\frac{3}{2}}}{\Gamma(\frac{5}{2})} \left(\frac{\exp(-s^2)}{32 \sqrt{1+s^2}} \frac{\sin u(s)+\sin v(s)}{16(s \exp(s^2)+1)} \right) ds, t \in [0, 1], \\ u(0) = \sqrt{3}, v(0) = \sqrt{2}, \end{cases} \tag{4.1}$$

We have $\alpha = \beta = \frac{1}{2}, \alpha_1 = \frac{3}{2}, \beta_1 = \frac{5}{2}, a = \sqrt{3}, b = \sqrt{2}, f_1(t, u, v) = \frac{\sin(u+v)}{18(\ln(t+1)+1)} + 6, f_2(t, u, v) = \frac{\sin u + \sin v}{16(t \exp(t^2)+6)}, \varphi_1(t) = \frac{\exp(-t)}{18(t+1)}$ and $\phi_1(t) = \frac{\exp(-t^2)}{32\sqrt{1+t^2}}$. Also, for $(u_1, v_1), (u_2, v_2) \in \mathbb{R}^2, t \in [0, 1]$, we have

$$|f_1(t, u_2, v_2) - f_1(t, u_1, v_1)| \leq \frac{1}{18} (|u_2 - u_1| + |v_2 - v_1|),$$

$$|f_2(t, u_2, v_2) - f_2(t, u_1, v_1)| \leq \frac{1}{16} (|u_2 - u_1| + |v_2 - v_1|),$$

$$|g_1(t, u_2, v_2) - g_1(t, u_1, v_1)| \leq \frac{1}{18} (|u_2 - u_1| + |v_2 - v_1|),$$

$$|h_1(t, u_2, v_2) - h_1(t, u_1, v_1)| \leq \frac{1}{16} (|u_2 - u_1| + |v_2 - v_1|).$$

Hence, $M_1 = 2.271, M_2 = 2.261, \mu_1 = \nu_1 = \frac{1}{18}, \mu_2 = \nu_2 = \frac{1}{16}, l_1 = m_1 = \frac{1}{18}, n_1 = k_1 = \frac{1}{16}$. Thus, we obtain $\bar{L} = \frac{1}{16}, \bar{L}(M_1 + M_2) = 0.283$. The conditions of the Theorem 3.1 hold. Therefore, the problem (4.1) has a unique solution on $[0, 1]$.

Example 4.2. Consider the following problem:

$$\left\{ \begin{array}{l} D^{\frac{3}{4}} u(t) = e^t \cos(u(t)v(t)) + \ln(t+4) \\ \quad + \int_0^t \frac{(t-s)^{\sqrt{11}-1}}{\Gamma(\sqrt{11})} \left[\frac{s}{e^s} \cos(su(s)v(s)) \right] ds, t \in [0, 1], \\ D^{\frac{5}{7}} v(t) = \sinh(-\pi t^2 |u(t)v(t)|) \\ \quad + \int_0^t \frac{(t-s)^{\sqrt{7}-1}}{\Gamma(\sqrt{7})} \left[\sqrt{s} \exp(-|u(s)| - |v(s)|) \right] ds, t \in [0, 1], \\ u(0) = \sqrt{2}, v(0) = \sqrt{5}. \end{array} \right. \quad (4.2)$$

For this example, we have $\alpha = \frac{3}{4}, \beta = \frac{5}{7}, a = \sqrt{2}, b = \sqrt{5}$, and for all $t \in [0, 1], \varphi_1(t) = \frac{t}{e^t}, \phi_1(t) = \sqrt{t}$, and for each $(u, v) \in \mathbb{R}^2$,

$$\begin{aligned} f_1(t, u, v) &= e^t \cos(uv) + \ln(t+4), \\ f_2(t, u, v) &= \sinh(-\pi t^2 |uv|). \end{aligned}$$

The conditions of Theorem 3.2 hold. Then (4.2) has at least one solution on $[0, 1]$.

References

- Abdellaoui, M. A., Z. Dahmani and M. Houas (2013). On some boundary value problems for coupled system of arbitrary order. *Indian Journal of industrial and applied mathematics* **4**(2), 180–188.
- Anber, A., S. Belarbi and Z. Dahmani (2013). New existence and uniqueness results for fractional differential equations. *An. St. Univ. Ovidius Constanta* **21**(3), 33–41.
- Bai, C. Z. and J. X. Fang (2004). The existence of a positive solution for a singular coupled system of nonlinear fractional differential equations. *Applied Mathematics and Computation* **150**(3), 611–621.
- Bengrine, M. E. and Z. Dahmani (2012). Boundary value problems for fractional differential equations. *Int. J. Open problems compt. Math* **5**(4), 7–15.

- Cui, Z., P. Yu and Z. Mao (2012). Existence of solutions for nonlocal boundary value problems of nonlinear fractional differential equations. *Advances in Dynamical Systems and Applications* **7**(1), 31–40.
- Gaber, M. and M. G. Brikaa (2012). Existence results for a couple system of nonlinear fractional differential equation with three point boundary conditions. *Journal of Fractional Calculus and Applications* **3**(21), 1–10.
- Gafiychuk, V., B. Datsko and V. Meleshko (2008). Mathematical modeling of time fractional reaction-diffusion systems. *Journal of Computational and Applied Mathematics* **220**(1-2), 215–225.
- Kilbas, A. A. and S. A. Marzan (2005). Nonlinear differential equation with the Caputo fractional derivative in the space of continuously differentiable functions. *Differ. Equ* **41**(1), 84–89.
- Lakshmikantham, V. and A. S. Vatsala (2008). Basic theory of fractional differential equations. *Nonlinear Anal* **69**(8), 2677–2682.
- Mainardi, F. (1997). Fractional calculus: Some basic problem in continuum and statistical mechanics. Fractals and fractional calculus in continuum mechanics. *Springer, Vienna*.
- Su, X. (2009). Boundary value problem for a coupled system of nonlinear fractional differential equations. *Applied Mathematics Letters* **22**(1), 64–69.
- Wang, J., H. Xiang and Z. Liu (2010). Positive solution to nonzero boundary values problem for a coupled system of nonlinear fractional differential equations. *International Journal of Differential Equations* **2010**, 1–12.
- Zhang, S. (2006). Positive solution for boundary value problem of nonlinear fractional differential equations. *Electron. J. Differential Equations* **2006**(36), 1–12.



On BV_σ I-convergent Sequence Spaces Defined by an Orlicz Function

Vakeel. A. Khan^{a,*}, Mohd Shafiq^a, Rami Kamel Ahmad Rababah^a

^aDepartment of Mathematics A.M.U, Aligarh-202002, India

Abstract

In this article we study ${}_0BV_\sigma^I(M)$, $BV_\sigma^I(M)$ and ${}_\infty BV_\sigma^I(M)$ sequence spaces with the help of BV_σ space see (Mursaleen, 1983b) and an Orlicz function M . we study some topological and algebraic properties of these spaces and prove some inclusion relations.

Keywords: bounded variation, invariant mean, σ -Bounded variation, ideal, filter, Orlicz function, I-convergence, I-null, solid space, sequence algebra, convergence free space.

2010 MSC: 41A10, 41A25, 41A36, 40A30.

1. Introduction and Preliminaries

Let \mathbb{N} , \mathbb{R} and \mathbb{C} be the sets of all natural, real and complex numbers respectively. We denote

$$\omega = \{x = (x_k) : x_k \in \mathbb{R} \text{ or } \mathbb{C}\}$$

the space of all real or complex sequences. Let ℓ_∞ , c and c_0 denote the Banach spaces of bounded, convergent and null sequences respectively with norm $\|x\| = \sup_k |x_k|$.

Let v be denote the space of sequences of bounded variation. That is,

$$v = \left\{ x = (x_k) : \sum_{k=0}^{\infty} |x_k - x_{k-1}| < \infty, x_{-1} = 0 \right\}, \quad (1.1)$$

v is a Banach Space normed by $\|x\| = \sum_{k=0}^{\infty} |x_k - x_{k-1}|$ (Mursaleen, 1983b). Let σ be an injective mapping of the set of the positive integers into itself having no finite orbits. A continuous linear functional ϕ on ℓ_∞ is said to be an invariant mean or σ -mean if and only if:

*Corresponding author

Email addresses: vakhanmaths@gmail.com (Vakeel. A. Khan), shafiqmaths7@gmail.com (Mohd Shafiq), rami215r@hotmail.com (Rami Kamel Ahmad Rababah)

1. $\phi(x) \geq 0$ where the sequence $x = (x_k)$ has $x_k \geq 0$ for all k ,
2. $\phi(e) = 1$ where $e = \{1, 1, 1, \dots\}$,
3. $\phi(x_{\sigma(n)}) = \phi(x)$ for all $x \in \ell_\infty$.

If $x = (x_k)$, write $Tx = (Tx_k) = (x_{\sigma(k)})$. It can be shown that

$$V_\sigma = \left\{ x = (x_k) : \lim_{m \rightarrow \infty} t_{m,k}(x) = L \text{ uniformly in } k, L = \sigma - \lim x \right\} \tag{1.2}$$

where $m \geq 0, k > 0$.

$$t_{m,k}(x) = \frac{x_k + x_{\sigma(k)} \dots + x_{\sigma^m(k)}}{m + 1} \text{ and } t_{-1,k} = 0, \tag{1.3}$$

where $\sigma_m(k)$ denote the m^{th} -iterate of $\sigma(k)$ at k . In case σ is the translation mapping, that is, $\sigma(k) = k + 1$, σ -mean is called a Banach limit (Banach, 1932) and V_σ , the set of bounded sequences of all whose invariant means are equal, is the set of almost convergent sequences. The special case of (1.2) in which $\sigma(k) = k + 1$ was given by (Lorentz, 1948), (Theorem 1), and that the general result can be proved in a similar way. It is familiar that a Banach limit extends the limit functional on c in the sense that

$$\phi(x) = \lim x, \text{ for all } x \in c. \tag{1.4}$$

Remark. In view of above discussion we have $c \subset V_\sigma$.

Theorem 1.1. A σ -mean extends the limit functional on c in the sense that $\phi(x) = \lim x$ for all $x \in c$ if and only if σ has no finite orbits. That is, if and only if for all $k \geq 0, j \geq 1, \sigma^j(k) \neq k$.

Put

$$\phi_{m,k}(x) = t_{m,k}(x) - t_{m-1,k}(x), \tag{1.5}$$

assuming that $t_{-1,k}(x) = 0$.

A straight forward calculation shows that (Mursaleen, 1983a)

$$\phi_{m,k}(x) = \left\{ \begin{array}{ll} \frac{1}{m(m+1)} \sum_{j=1}^m j(x_{\sigma^j(k)} - x_{\sigma^{j-1}(k)}), & \text{if } (m \geq 1), \\ x_k & \text{if } (m = 0) \end{array} \right\}. \tag{1.6}$$

For any sequence x, y and scalar λ , we have $\phi_{m,k}(x + y) = \phi_{m,k}(x) + \phi_{m,k}(y)$ and $\phi_{m,k}(\lambda x) = \lambda \phi_{m,k}(x)$.

Definition 1.1. A sequence $x \in \ell_\infty$ is of σ -bounded variation if and only if

- (i) $\sum_{m=0}^\infty |\phi_{m,k}(x)|$ converges uniformly in k .
- (ii) $\lim_{m \rightarrow \infty} t_{m,k}(x)$, which must exist, should take the same value for all k .

Subsequently invariant means have been studied by (Mursaleen, 1983b,a; Ahmad & Mursaleen, 1986; Raimi, 1963; Khan & Ebadullah, 2013, 2012; Schafer, 1972) and many others. (Mursaleen, 1983b) defined the sequence space BV_σ , the space of all sequences of σ -bounded variation as $BV_\sigma = \{x \in \ell_\infty : \sum_m |\phi_{m,k}(x)| < \infty, \text{ uniformly in } k\}$.

Theorem 1.2. (Fast, 1951) BV_σ is a Banach space normed by $\|x\| = \sup_k \sum |\phi_{m,k}(x)|$.

Definition 1.2. (see[23]) A function $M : [0, \infty) \rightarrow [0, \infty)$ is said to be an Orlicz function if it satisfies the following conditions;

- (i) M is continuous, convex and non-decreasing,
- (ii) $M(0) = 0, M(x) > 0$ and $M(x) \rightarrow \infty$ as $x \rightarrow \infty$.

Remark. (see (Tripathy & Hazarika, 2011)) If the convexity of an Orlicz function is replaced by $M(x + y) \leq M(x) + M(y)$, then this function is called modulus function.

Remark. If M is an Orlicz function, then $M(\lambda x) \leq \lambda M(x)$ for all λ with $0 < \lambda < 1$.

An Orlicz function M is said to satisfy Δ_2 – Condition for all values of u if there exists a constant $K > 0$ such that $M(Lu) \leq KLM(u)$ for all values of $L > 1$ (see (Tripathy & Hazarika, 2011)).

(Lindenstrauss & Tzafriri, 1971) used the idea of an Orlicz function to construct the sequence space $\ell_M = \{x \in \omega : \sum_{k=1}^{\infty} M(\frac{|x_k|}{\rho}) < \infty, \text{ for some } \rho > 0\}$. The space ℓ_M becomes a Banach space with the norm

$$\|x\| = \inf\{\rho > 0 : \sum_{k=1}^{\infty} M(\frac{|x_k|}{\rho}) \leq 1\}, \quad (1.7)$$

which is called an Orlicz sequence space. The space ℓ_M is closely related to the space ℓ_p which is an Orlicz sequence space with $M(t) = t^p$ for $1 \leq p < \infty$.

Later on, some Orlicz sequence spaces were investigated by (Hazarika & Esi, 2013; Maddox, 1970; Parshar & Choudhary, 1994; Bhardwaj & Singh, 2000; Et, 2001; Tripathy & Hazarika, 2011) and many others.

Initially, as a generalization of statistical convergence (Fridy, 1985), the notation of ideal convergence (I-convergence) was introduced and studied by (P. Kostyrko & Wilczyński, 2000). Later on, it was studied by (Khan & Ebadullah, 2013), (Hazarika & Esi, 2013; T. Šalát & Ziman, 2004, 2005) and many others.

Here we give some preliminaries about the notion of I-convergence.

Definition 1.3. A sequence $x = (x_k) \in \omega$ is said to be statistically convergent to a limit $L \in \mathbb{C}$ if for every $\epsilon > 0$, we have $\lim_k \frac{1}{k} |\{n \in \mathbb{N} : |x_n - L| \geq \epsilon, n \leq k\}| = 0$, where vertical lines denote the cardinality of the enclosed set.

Definition 1.4. Let \mathbb{N} be the set of natural numbers. Then a family of sets $I \subseteq 2^{\mathbb{N}}$ (power set of \mathbb{N}) is said to be an ideal if:

- 1) I is additive i.e $\forall A, B \in I \Rightarrow A \cup B \in I$,
- 2) I is hereditary i.e $\forall A \in I \text{ and } B \subseteq A \Rightarrow B \in I$.

Definition 1.5. A non-empty family of sets $\mathfrak{F}(I) \subseteq 2^{\mathbb{N}}$ is said to be filter on \mathbb{N} if and only if

- 1) $\Phi \notin \mathfrak{F}(I)$,
- 2) $\forall A, B \in \mathfrak{F}(I)$ we have $A \cap B \in \mathfrak{F}(I)$,
- 3) $\forall A \in \mathfrak{F}(I)$ and $A \subseteq B \Rightarrow B \in \mathfrak{F}(I)$.

Definition 1.6. An Ideal $I \subseteq 2^{\mathbb{N}}$ is called non-trivial if $I \neq 2^{\mathbb{N}}$.

Definition 1.7. A non-trivial ideal $I \subseteq 2^{\mathbb{N}}$ is called admissible if $\{\{x\} : x \in \mathbb{N}\} \subseteq I$.

Definition 1.8. A non-trivial ideal I is maximal if there cannot exist any non-trivial ideal $J \neq I$ containing I as a subset.

Definition 1.9. For each ideal I , there is a filter $\mathfrak{f}(I)$ corresponding to I .
i.e $\mathfrak{f}(I) = \{K \subseteq \mathbb{N} : K^c \in I\}$, where $K^c = \mathbb{N} \setminus K$.

Definition 1.10. A sequence $x = (x_k) \in \omega$ is said to be I -convergent to a number L if for every $\epsilon > 0$, the set $\{k \in \mathbb{N} : |x_k - L| \geq \epsilon\} \in I$.
In this case, we write $I - \lim x_k = L$.

Definition 1.11. A sequence $x = (x_k) \in \omega$ is said to be I -null if $L = 0$. In this case, we write $I - \lim x_k = 0$.

Definition 1.12. A sequence $x = (x_k) \in \omega$ is said to be I -cauchy if for every $\epsilon > 0$ there exists a number $m = m(\epsilon)$ such that $\{k \in \mathbb{N} : |x_k - x_m| \geq \epsilon\} \in I$.

Definition 1.13. A sequence space E is said to be solid(normal) if $(\alpha_k x_k) \in E$ whenever $(x_k) \in E$ and for any sequence (α_k) of scalars with $|\alpha_k| \leq 1$, for all $k \in \mathbb{N}$.

Definition 1.14. A sequence space E is said to be symmetric if $(x_{\pi(k)}) \in E$ whenever $(x_k) \in E$. where π is a permutation on \mathbb{N} .

Definition 1.15. A sequence space E is said to be sequence algebra if $(x_k) * (y_k) = (x_k \cdot y_k) \in E$ whenever $(x_k), (y_k) \in E$.

Definition 1.16. A sequence space E is said to be convergence free if $(y_k) \in E$ whenever $(x_k) \in E$ and $x_k = 0$ implies $y_k = 0$, for all k .

Definition 1.17. Let $K = \{k_1 < k_2 < k_3 < k_4 < k_5 \dots\} \subset \mathbb{N}$ and E be a Sequence space. A K -step space of E is a sequence space $\lambda_K^E = \{(x_{k_n}) \in \omega : (x_k) \in E\}$.

Definition 1.18. A canonical pre-image of a sequence $(x_{k_n}) \in \lambda_K^E$ is a sequence $(y_k) \in \omega$ defined by

$$y_k = \begin{cases} x_k, & \text{if } k \in K, \\ 0, & \text{otherwise.} \end{cases}$$

A canonical preimage of a step space λ_K^E is a set of preimages all elements in λ_K^E . i.e. y is in the canonical preimage of λ_K^E iff y is the canonical preimage of some $x \in \lambda_K^E$.

Definition 1.19. A sequence space E is said to be monotone if it contains the canonical preimages of its step space.

Remark. If $I = I_f$, the class of all finite subsets of \mathbb{N} . Then, I is an admissible ideal in \mathbb{N} and I_f convergence coincides with the usual convergence.

Definition 1.20. If $I = I_\delta = \{A \subseteq \mathbb{N} : \delta(A) = 0\}$. Then, I is an admissible ideal in N and we call the I_δ -convergence as the logarithmic statistical convergence.

Definition 1.21. If $I = I_d = \{A \subseteq \mathbb{N} : d(A) = 0\}$. Then, I is an admissible ideal in \mathbb{N} and we call the I_d -convergence as the asymptotic statistical convergence.

Remark. If $I_\delta - \lim x_k = l$, then $I_d - \lim x_k = l$.

The following lemmas remained an important tool for the establishment of some results of this article.

Lemma 1.1. (Tripathy & Hazarika, 2011). Every solid space is monotone.

Lemma 1.2. Let $K \in \mathfrak{I}(I)$ and $M \subseteq \mathbb{N}$. If $M \notin I$, then $M \cap K \notin I$.

Lemma 1.3. If $I \subseteq 2^{\mathbb{N}}$ and $M \subseteq N$. If $M \notin I$, then $M \cap N \notin I$.

2. Main Results

Recently (Khan & Ebadullah, 2012) introduced and studied the following sequence space. For $m \geq 0$

$$BV_\sigma^I = \left\{ x = (x_k) \in \omega : \{k \in \mathbb{N} : |\phi_{m,k}(x) - L| \geq \epsilon\} \in I, \text{ for some } L \in \mathbb{C} \right\}. \quad (2.1)$$

In this article we introduce the following sequence spaces. For $m \geq 0$

$$BV_\sigma^I(M) = \left\{ x = (x_k) \in \omega : I - \lim M \left(\frac{|\phi_{m,k}(x) - L|}{\rho} \right) = 0, \text{ for some } L \in \mathbb{C}, \rho > 0 \right\}, \quad (2.2)$$

$${}_0BV_\sigma^I(M) = \left\{ x = (x_k) \in \omega : I - \lim M \left(\frac{|\phi_{m,k}(x)|}{\rho} \right) = 0, \rho > 0 \right\}, \quad (2.3)$$

$${}_\infty BV_\sigma^I(M) = \left\{ x = (x_k) \in \omega : \{k \in \mathbb{N} : \exists K > 0 \text{ s.t. } M \left(\frac{|\phi_{m,k}(x)|}{\rho} \right) \geq K\} \in I, \rho > 0 \right\}, \quad (2.4)$$

$${}_\infty BV_\sigma(M) = \left\{ x = (x_k) \in \omega : \sup M \left(\frac{|\phi_{m,k}(x)|}{\rho} \right) < \infty, \rho > 0 \right\}. \quad (2.5)$$

We also denote $\mathcal{M}_{BV_\sigma}^I(M) = BV_\sigma^I(M) \cap {}_\infty BV_\sigma(M)$ and ${}_0\mathcal{M}_{BV_\sigma}^I(M) = {}_0BV_\sigma^I(M) \cap {}_\infty BV_\sigma(M)$.

Throughout the article, if required, we denote $\phi_{m,k}(x) = x'$, $\phi_{m,k}(y) = y'$ and $\phi_{m,k}(z) = z'$ where x, y, z are $(x_k), (y_k)$ and (z_k) respectively.

Theorem 2.1. For any Orlicz function M , the classes of sequence ${}_0BV_\sigma^I(M)$, $BV_\sigma^I(M)$, ${}_0\mathcal{M}_{BV_\sigma}^I(M)$ and $\mathcal{M}_{BV_\sigma}^I(M)$ are the linear spaces.

Proof. We shall prove the result for the space $BV_{\sigma}^I(M)$, others will follow similarly.

For, let $x = (x_k), y = (y_k) \in BV_{\sigma}^I(M)$ be any two arbitrary elements and let α, β are scalars.

Now, since $(x_k), (y_k) \in BV_{\sigma}^I(M) \Rightarrow \exists$ some positive numbers $L_1, L_2 \in \mathbb{C}$ and $\rho_1, \rho_2 > 0$ such that

$$I - \lim_k M\left(\frac{|\phi_{m,k}(x) - L_1|}{\rho_1}\right) = 0, \tag{2.6}$$

$$I - \lim_k M\left(\frac{|\phi_{m,k}(y) - L_2|}{\rho_2}\right) = 0 \tag{2.7}$$

\Rightarrow for any given $\epsilon > 0$, the sets

$$A_1 = \left\{k \in \mathbb{N} : M\left(\frac{|\phi_{m,k}(x) - L_1|}{\rho_1}\right) > \frac{\epsilon}{2}\right\} \in I, \tag{2.8}$$

$$A_2 = \left\{k \in \mathbb{N} : M\left(\frac{|\phi_{m,k}(y) - L_2|}{\rho_2}\right) > \frac{\epsilon}{2}\right\} \in I. \tag{2.9}$$

Let

$$\rho_3 = \max\{2|\alpha|\rho_1, 2|\beta|\rho_2\}. \tag{2.10}$$

Since, M is non-decreasing and convex function, we have

$$M\left(\frac{|\alpha x'_k + \beta y'_k - (\alpha L_1 + \beta L_2)|}{\rho_3}\right) \leq M\left(\frac{|\alpha| |x'_k - L_1|}{\rho_3}\right) + M\left(\frac{|\beta| |y'_k - L_2|}{\rho_3}\right) \leq M\left(\frac{|x'_k - L_1|}{\rho_1}\right) + M\left(\frac{|y'_k - L_2|}{\rho_2}\right). \tag{2.11}$$

Therefore, from (2.8), (2.9) and (2.11), we have $\left\{k \in \mathbb{N} : M\left(\frac{|\alpha x'_k + \beta y'_k - (\alpha L_1 + \beta L_2)|}{\rho_3}\right) > \epsilon\right\} \subseteq A_1 \cup A_2 \in I$

implies that $\left\{k \in \mathbb{N} : M\left(\frac{|\alpha x'_k + \beta y'_k - (\alpha L_1 + \beta L_2)|}{\rho_3}\right) > \epsilon\right\} \in I$. That is, $I - \lim M\left(\frac{|\alpha x'_k + \beta y'_k - (\alpha L_1 + \beta L_2)|}{\rho_3}\right) = 0$. Thus,

$\alpha x_k + \beta y_k \in BV_{\sigma}^I(M)$. But $(x_k), (y_k) \in BV_{\sigma}^I(M)$ are the arbitrary elements. Therefore, $\alpha x_k + \beta y_k \in BV_{\sigma}^I(M)$, for all $(x_k), (y_k) \in BV_{\sigma}^I(M)$ and for all scalars α, β . Hence, $BV_{\sigma}^I(M)$ is linear. \square

Theorem 2.2. Let M_1 and M_2 be two Orlicz functions and satisfying Δ_2 – Condition, then

(a) $\mathcal{X}(M_2) \subseteq \mathcal{X}(M_1 M_2)$,

(b) $\mathcal{X}(M_1) \cap \mathcal{X}(M_2) \subseteq \mathcal{X}(M_1 + M_2)$ for $\mathcal{X} = {}_0BV_{\sigma}^I, BV_{\sigma}^I, {}_0\mathcal{M}_{BV_{\sigma}^I}^I$ and $\mathcal{M}_{BV_{\sigma}^I}^I$.

Proof. (a) Let $x = (x_k) \in {}_0BV_{\sigma}^I(M_2)$ be any arbitrary element $\Rightarrow \exists \rho > 0$ such that

$$I - \lim M_2\left(\frac{|\phi_{m,k}(x)|}{\rho}\right) = 0, \tag{2.12}$$

i.e.

$$I - \lim M_2\left(\frac{|x'_k|}{\rho}\right) = 0. \tag{2.13}$$

Let $\epsilon > 0$ and choose δ with $0 < \delta < 1$ such that $M_1(t) < \epsilon$, $0 \leq t \leq \delta$. Let us write $y_k = M_2\left(\frac{|x'_k|}{\rho}\right)$ and consider

$$\lim_k M_1(y_k) = \lim_{y_k \leq \delta, k \in \mathbb{N}} M_1(y_k) + \lim_{y_k > \delta, k \in \mathbb{N}} M_1(y_k). \tag{2.14}$$

Now, since M_1 is an Orlicz function, we have $M_1(\lambda x) \leq \lambda M_1(x)$ for all λ with $0 < \lambda < 1$. Therefore,

$$\lim_{y_k \leq \delta, k \in \mathbb{N}} M_1(y_k) \leq M_1(2) \lim_{y_k \leq \delta, k \in \mathbb{N}} (y_k). \tag{2.15}$$

For $y_k > \delta$, we have $y_k < \frac{y_k}{\delta} < 1 + \frac{y_k}{\delta}$. Now, since M_1 is non-decreasing and convex, it follows that

$$M_1(y_k) < M_1\left(1 + \frac{y_k}{\delta}\right) < \frac{1}{2}M_1(2) + \frac{1}{2}M_1\left(\frac{2y_k}{\delta}\right). \tag{2.16}$$

Again, since M_1 satisfies Δ_2 – Condition, we have $M_1(y_k) < \frac{1}{2}K\left(\frac{y_k}{\delta}\right)M_1(2) + \frac{1}{2}K\left(\frac{y_k}{\delta}\right)M_1(2)$. Thus, $M_1(y_k) < K\left(\frac{y_k}{\delta}\right)M_1(2)$. Hence,

$$\lim_{y_k > \delta, k \in \mathbb{N}} M_1(y_k) \leq \max\{1, K\delta^{-1}M_1(2)\} \lim_{y_k > \delta, k \in \mathbb{N}} (y_k). \tag{2.17}$$

Therefore, from (2.12), (2.13) and (2.14), we have $I\text{-}\lim_k M_1(y_k) = 0$, i.e. $I\text{-}\lim_k M_1 M_2\left(\frac{|\phi_{m,k}(x)|}{\rho}\right) = 0$, implies that $(x_k) \in {}_0BV_\sigma^I(M_1 M_2)$. Thus, ${}_0BV_\sigma^I(M_2) \subseteq {}_0BV_\sigma^I(M_1 M_2)$. Hence, $\mathcal{X}(M_2) \subseteq \mathcal{X}(M_1 M_2)$ for $\mathcal{X} = {}_0BV_\sigma^I$. For $\mathcal{X} = BV_\sigma^I, \mathcal{X} = {}_0M_{BV_\sigma}^I$ and $\mathcal{X} = \mathcal{M}_{BV_\sigma^I}$ the inclusions can be established similarly.

(b). Let $x = (x_k) \in {}_0BV_\sigma^I(M_1) \cap {}_0BV_\sigma^I(M_2)$. Let $\epsilon > 0$ be given. Then there exists $\rho > 0$ such that the sets $I\text{-}\lim M_1\left(\frac{|\phi_{m,k}(x)|}{\rho}\right) = 0$ and $I\text{-}\lim M_2\left(\frac{|\phi_{m,k}(x)|}{\rho}\right) = 0$. Therefore, $I\text{-}\lim M_1 + M_2\left(\frac{|\phi_{m,k}(x)|}{\rho}\right) = I\text{-}\lim M_1\left(\frac{|\phi_{m,k}(x)|}{\rho}\right) + I\text{-}\lim M_2\left(\frac{|\phi_{m,k}(x)|}{\rho}\right) = 0$. Thus, $x = (x_k) \in {}_0BV_\sigma^I(M_1 + M_2)$. Hence, ${}_0BV_\sigma^I(M_1) \cap {}_0BV_\sigma^I(M_2) \subseteq {}_0BV_\sigma^I(M_1 + M_2)$. For $\mathcal{X} = BV_\sigma^I, \mathcal{X} = {}_0M_{BV_\sigma}^I$ and $\mathcal{X} = \mathcal{M}_{BV_\sigma^I}^I$ the inclusions are similar. \square

For $M_2(x) = (x)$ and $M_1(x) = M(x)$, $\forall x \in [0, \infty)$, we have the following corollary.

Corollary 2.1. $\mathcal{X} \subseteq \mathcal{X}(M)$ for $\mathcal{X} = {}_0BV_\sigma^I, BV_\sigma^I, {}_0M_{BV_\sigma}^I$ and $\mathcal{M}_{BV_\sigma^I}^I$.

Theorem 2.3. For any orlicz function M , the spaces ${}_0BV_\sigma^I(M)$ and ${}_0M_{BV_\sigma}^I$ are solid and monotone.

Proof. Here we consider ${}_0BV_\sigma^I(M)$ and for ${}_0M_{BV_\sigma}^I$ the proof shall be similar. For, let $(x_k) \in {}_0BV_\sigma^I(M)$ be any arbitrary element. $\Rightarrow \exists \rho > 0$ such that $I\text{-}\lim_k M\left(\frac{|\phi_{m,k}(x)|}{\rho}\right) = 0$. Let (α_k) be a sequence of scalars such that $|\alpha_k| \leq 1$, for all $k \in \mathbb{N}$.

Now, since M is an Orlicz function. Therefore,

$$\begin{aligned} M\left(\frac{|\alpha_k \phi_{m,k}(x)|}{\rho}\right) &\leq |\alpha_k| M\left(\frac{|\phi_{m,k}(x)|}{\rho}\right) \leq M\left(\frac{|\phi_{m,k}(x)|}{\rho}\right) \\ \Rightarrow M\left(\frac{|\alpha_k \phi_{m,k}(x)|}{\rho}\right) &\leq M\left(\frac{|\phi_{m,k}(x)|}{\rho}\right), \text{ for all } k \in \mathbb{N}, \end{aligned}$$

implies that $I - \lim_k M\left(\frac{|\alpha_k \phi_{m,k}(x)|}{\rho}\right) = 0$.

Thus, $(\alpha_k x_k) \in {}_0BV_\sigma^I(M)$. Hence ${}_0BV_\sigma^I(M)$ is solid. Therefore, by lemma 1.1, ${}_0BV_\sigma^I(M)$ is monotone. Hence the result. \square

Theorem 2.4. For any orlicz function M , the spaces $BV_\sigma^I(M)$ and $\mathcal{M}_{BV_\sigma}^I$ are neither solid nor monotone in general.

Proof. Here we give counter example for the establishment of this result. For, let us consider $I = I_f$ and $M(x) = x$, for all $x \in [0, \infty)$. Consider, the K -step space $B_K(M)$ of $B(M)$ as follows. Let $(x_k) \in B(M)$ and $(y_k) \in B_K(M)$ be such that

$$y_k = \begin{cases} x_k, & \text{if } k \text{ is even,} \\ 0, & \text{otherwise.} \end{cases}$$

Consider the sequence (x_k) defined as $x_k = 1$, for all $k \in \mathbb{N}$, then $x_k \in BV_\sigma^I(M)$ and $\mathcal{M}_{BV_\sigma}^I$ but its K -step space pre-image does not belong to $BV_\sigma^I(M)$ and $\mathcal{M}_{BV_\sigma}^I$. Thus, $BV_\sigma^I(M)$ and $\mathcal{M}_{BV_\sigma}^I$ are not monotone and hence by lemma(I) they are not solid. \square

Theorem 2.5. For an Orlicz function M , the spaces ${}_0BV_\sigma^I(M)$ and $BV_\sigma^I(M)$ are not convergence free.

Proof. Let $I = I_f$ and $M(x) = x$ for all $x \in [0, \infty)$. Consider the sequences (x_k) and (y_k) defined as follows.

$$x_k = \frac{1}{k} \text{ and } y_k = k, \text{ for all } k \in \mathbb{N}.$$

Then, (x_k) belongs to both ${}_0BV_\sigma^I(M)$ and $BV_\sigma^I(M)$ but (y_k) does not belongs to both ${}_0BV_\sigma^I(M)$ and $BV_\sigma^I(M)$.

Hence, the spaces ${}_0BV_\sigma^I(M)$ and $BV_\sigma^I(M)$ are not convergence free. \square

Theorem 2.6. For an Orlicz function M , the spaces ${}_0BV_\sigma^I(M)$ and $BV_\sigma^I(M)$ are sequence algebra.

Proof. Here we consider ${}_0BV_\sigma^I(M)$. For the other one, result is similar.

Let $x = (x_k), y = (y_k) \in {}_0BV_\sigma^I(M)$ be any two arbitrary elements.

$\Rightarrow \exists \rho_1, \rho_2 > 0$ such that

$$I - \lim_k M\left(\frac{|\phi_{m,k}(x)|}{\rho_1}\right) = 0$$

and

$$I - \lim_k M\left(\frac{|\phi_{m,k}(y)|}{\rho_2}\right) = 0.$$

Let $\rho = \rho_1 \rho_2 > 0$. Then, it is obvious that $I - \lim_k M\left(\frac{|\phi_{m,k}(x)\phi_{m,k}(y)|}{\rho}\right) = 0$ implies that $(x_k \cdot y_k) = (x_k y_k) \in {}_0BV_\sigma^I(M)$. Hence, ${}_0BV_\sigma^I(M)$ is a Sequence algebra. \square

Theorem 2.7. Let M be an Orlicz function. Then, ${}_0BV_\sigma^I(M) \subseteq BV_\sigma^I(M) \subseteq {}_\infty BV_\sigma^I(M)$.

Proof. Let M be an Orlicz function. Then, we have to show that ${}_0BV_\sigma^I(M) \subseteq BV_\sigma^I(M) \subseteq {}_\infty BV_\sigma^I(M)$. Firstly, ${}_0BV_\sigma^I(M) \subseteq BV_\sigma^I(M)$ is obvious.

Now, let $x = (x_k) \in BV_\sigma^I(M)$ be any arbitrary element $\Rightarrow \exists \rho > 0$ such that $I\text{-}\lim_k M\left(\frac{|\phi_{m,k}(x)-L|}{\rho}\right) = 0$ for some $L \in \mathbb{N}$.

Now, $M\left(\frac{|\phi_{m,k}(x)|}{2\rho}\right) \leq \frac{1}{2}M\left(\frac{|\phi_{m,k}(x)-L|}{\rho}\right) + \frac{1}{2}M\left(\frac{|L|}{\rho}\right)$. Taking supremum over k to both sides, we have $x = (x_k) \in {}_\infty BV_\sigma^I(M)$. Hence, ${}_0BV_\sigma^I(M) \subseteq BV_\sigma^I(M) \subseteq {}_\infty BV_\sigma^I(M)$. \square

Acknowledgments: The authors would like to record their gratitude to the reviewers for their careful reading and making some useful corrections which improved the presentation of the paper.

References

- Ahmad, Z. U and M. Mursaleen (1986). An application of Banach limits. *Proc. Amer. Math. Soc.*
- Banach, S. (1932). Théorie des opérations linéaires. *Warszawa*.
- Bhardwaj, V. K. and N. Singh (2000). Some sequence spaces defined by Orlicz functions. *Demonstratio Math.* **33**(3), 571–582.
- Et, M. (2001). On some new Orlicz spaces. *J. Analysis* **9**, 21–28.
- Fast, H. (1951). Sur la convergence statistique. **2**, 241–244.
- Fridy, J. A. (1985). On statistical convergence. *Analysis* **5**, 301–313.
- Hazarika, B. and A. Esi (2013). Some I-convergent generalized difference lacunary double sequence spaces defined by Orlicz function. *Acta Scientiarum. Technology* **35**(3), 527–537.
- Khan, V. A. and K. Ebadullah (2012). On a new I-convergent sequence space. *Analysis* **32**, 199–208.
- Khan, V. A. and K. Ebadullah (2013). On some new I-convergent sequence space. *Mathematics, Aeterna* **3**(2), 151–159.
- Lindenstrauss, J. and L. Tzafriri (1971). On Orlicz sequence spaces. *Israel J. Math.* **101**, 379–390.
- Lorentz, G. G. (1948). A contribution to the theory of divergent series. *Acta Math.* **80**(6), 167–190.
- Maddox, I. J. (1970). *Elements of functional analysis*. Cambridge University Press.
- Mursaleen, M. (1983a). Matrix transformation between some new sequence spaces. *Houston J. Math.*
- Mursaleen, M. (1983b). On some new invariant matrix methods of summability. *Quart. J. Math. Oxford* **9**, 77–86.
- P. Kostyrko, T. Šalát and W. Wilczyński (2000). I-convergence. *Raal Analysis Analysis Exchange* **26**(2), 669–686.
- Parshar, S. D. and B. Choudhary (1994). Sequence spaces defined by Orlicz function. *Indian J. Pure Appl. Math.* **25**, 419–428.
- Raimi, R. A. (1963). Invariant means and invariant matrix method summability. *Duke J. Math* **30**, 81–94.
- Schafer, P. (1972). Infinite matrices and invariant means. *Proc. Amer. soc.* **36**, 104–110.
- T. Šalát, B.C. Tripathy and M. Ziman (2004). On some properties of I-convergence. *Tatra Mt. Math. Publ.* **28**, 279–286.
- T. Šalát, B.C. Tripathy and M. Ziman (2005). On I-convergence field. *Ital. J. Pure Appl. Math.* **17**, 45–54.
- Tripathy, B.C. and B. Hazarika (2011). Some I-convergent sequence spaces defined by Orlicz function. *Acta Mathematicae Applicatae Sinica* **27**(1), 149–154.



Properties of Stabilizing Computations

Mark Burgin^a

^a*University of California, Los Angeles 405 Hilgard Ave. Los Angeles, CA 90095*

Abstract

Models play an important role in the development of computer science and information technology applications. Turing machine is one of the most popular model of computing devices and computations. This model, or more exactly, a family of models, provides means for exploration of capabilities of information technology. However, a Turing machine stops after giving a result. In contrast to this, computers, networks and their software, such as an operating system, very often work without stopping but give various results. There are different modes of such functioning and Turing machines do not provide adequate models for these processes. One of the closest to halting computation is stabilizing computation when the output has to stabilize in order to become the result of a computational process. Such stabilizing computations are modeled by inductive Turing machines. In comparison with Turing machines, inductive Turing machines represent the next step in the development of computer science providing better models for contemporary computers and computer networks. At the same time, inductive Turing machines reflect pivotal traits of stabilizing computational processes. In this paper, we study relations between different modes of inductive Turing machines functioning. In particular, it is demonstrated that acceptance by output stabilizing and acceptance by state stabilizing are linguistically equivalent.

Keywords: computation, stability, Turing machine, inductive Turing machine, acceptance, mode of computation, equivalence.

2010 MSC: 68Q05.

2012 CCS: theory of computation, models of computation.

1. Introduction

Computer science studies computations by means of theoretical models. One of the most popular theoretical models of computation is Turing machine. It is central in computer science and in many applications, especially, when it is necessary to prove impossibility of an algorithmic solution to a problem (Rogers, 1987). The pivotal feature of a Turing machine is the necessity to stop after giving a result because all subsequent machine operations become superfluous.

However, computers, networks and their software, such as an operating system, very often work without stopping but give various results. There are different modes of such functioning and Turing machines do not provide adequate models for these processes (Burgin, 2005a). Stabilizing computation is one of the closest to halting computation computational modes when the output has to stabilize in order to become the result of a computational process. Such stabilizing computations

are efficiently modeled by inductive Turing machines (Burgin & Debnath, 2004, 2005; Burgin, 2005a, 2006; Burgin & Gupta, 2012).

In comparison with Turing machines, inductive Turing machines represent the next step in the development of computer science providing better modeling tools for contemporary computers and computer networks (Burgin, 2005a). In particular, even simple inductive Turing machines and other inductive Turing machines of the first order can solve the Halting Problem for Turing machines, while inductive Turing machines of higher orders can generate and decide the whole arithmetical hierarchy as it is proved in (Burgin, 2003). Even more, unrestricted inductive Turing machines with a structured memory have the same computing power as Turing machines with oracles (Burgin, 2005a). In addition, inductive Turing machines allow decreasing time of computations (Burgin, 1999). Being more powerful, inductive Turing machines allow essential reduction of Kolmogorov (algorithmic) complexity of finite objects (Burgin, 2004), as well as algorithmic complexity of mathematical and computational problems (Burgin, 2010a). It is also important that in contrast to Turing machines, which can work only with words (Turing machines with one one-dimensional tape), with finite systems of words (Turing machines with several one-dimensional tapes) and with arrays (Turing machines with multidimensional tapes), inductive Turing machines can work not only with finite and infinite words, systems of words and multidimensional arrays but also with more sophisticated data structures, such as graphs, functions, hierarchical structures and chains of named sets or named data.

Inductive Turing machines have found applications in algorithmic information theory and complexity studies (Burgin, 2004, 2007, 2010a), software testing (Burgin & Debnath, 2009; Burgin *et al.*, 2009), high performance computing (Burgin, 1999), machine learning (Burgin & Klinger, 2004), software engineering (Burgin & Debnath, 2004, 2005), computer networks (Burgin, 2006; Burgin & Gupta, 2012) and evolutionary computations (Burgin & Eberbach, 2008, 2009b,a, 2010, 2012). For instance, inductive Turing machines can perform all types of machine learning - TxtEx-learning, TxtFin-learning, TxtBC-learning, and TxtEx*-learning, (Beros, 2013). While the traditional approach to machine learning models learning processes using functions, e.g., limit partial recursive functions (Gold, 1967), inductive Turing machines are automata, which can compute values of the modeling functions.

An important area of tentative application of inductive Turing machines and other super-recursive algorithms is software development and maintenance. As Călinescu, et al, (Calinescu *et al.*, 2013) write, modern software systems are often complex, inevitably distributed, and operate in heterogeneous and highly dynamic environments. Examples of such systems include those from the service-oriented, cloud computing, and pervasive computing domains. In these domains, continuous change is the norm and therefore the software must also change accordingly. In many cases, the software is required to self-react by adapting its behavior dynamically, in order to ensure required levels of service quality in changing environments. As a result, conventional recursive algorithms, such as Turing machines, cannot provide efficient means for modeling software functioning and behavior. This can be achieved only by utilization of inductive Turing machines and other super-recursive algorithms. Thus, better knowledge of inductive Turing machines properties and regularities of their behavior allows their better utilization and application of these models of computation and computer systems.

The goal of this paper is to study inductive Turing machines as models of real computing

devices, functioning of which often results in stabilizing computations. Note that stability is an important property of a computing device, as well as of computational processes. By definition, inductive Turing machines give results if and only if their computational process stabilizes (Burgin, 2005a).

Each real computing device, e.g., a computer, has three components: an input device (devices), an output device (devices) and a processor or a multiprocessor, which consists of several processors. Consequently, there are three modes of component functioning: the input mode, output mode and processing mode. Together they form the functioning mode of the computing device. For instance, the processing mode of an automaton can be acceptance (of the input), decision (about the input) or computation (of the final result). The output mode of a pushdown automaton can be acceptance by final state or acceptance by empty stack (Hopcroft *et al.*, 2001). The output mode of a Turing machine can be acceptance by final state or acceptance by halting (Hopcroft *et al.*, 2001). An inductive Turing machine can process information in a recursive mode or in the inductive mode (Burgin, 2005a). Computer scientists are usually interested in equivalence of different computational modes as it allows them to use the most appropriate mode for solving a given problem without loss of generality.

There are different types of equivalence: linguistic equivalence, functional equivalence, process equivalence, etc. (cf. (Burgin, 2010b)). Here we study the classical case of equivalence called linguistic equivalence.

We remind that two automata (computing devices) are linguistically equivalent if they have the same language (Burgin, 2010b). Note that it may be linguistic equivalence with respect to computation when the language of the automaton A is the language computed by A or it may be linguistic equivalence with respect to acceptance when the language of the automaton A is the language accepted by A.

We remind that two classes of automata are linguistically equivalent if they have the same classes of languages and two modes of functioning are linguistically equivalent if the classes of automata working in these modes are linguistically equivalent (Burgin, 2010b). As separate automata, classes of automata may be linguistically equivalent with respect to computation or with respect to acceptance. One of the basic results of the theory of pushdown automata is the statement that acceptance by final state is linguistically equivalent to acceptance by empty stack ((Hopcroft *et al.*, 2001) Section 6.2).

One of the basic results of the theory of Turing machines and recursive computations is the statement that acceptance by final state is linguistically equivalent to acceptance by halting ((Burgin, 2005a), Chapter 2).

The goal of this paper is to analyze different modes of inductive Turing machine functioning, finding whether similar results are true for these modes. Note that inductive Turing machines have much more modes of functioning than Turing machines. In Section 2, we remind some basic concepts and constructions from the theory of inductive Turing machines and stabilizing computations. In Section 3, we demonstrate that some key modes of inductive Turing machine functioning are linguistically equivalent.

2. Inductive Turing machines as models of stabilizing computations

To understand how inductive Turing machines model stabilizing computations, we need to know the hardware structure and characteristics of inductive Turing machine functioning in general and of the simple inductive Turing machine functioning, in particular, making the emphasis on work with finite words in some alphabet (Burgin, 2005a).

An inductive Turing machine M hardware consists of three abstract devices: a *control device* A , which is a finite automaton and controls performance of M ; a *processor or operating device* H , which corresponds to one or several *heads* of a conventional Turing machine; and the *memory* E , which corresponds to the *tape* or tapes of a conventional Turing machine. The memory E of the simplest inductive Turing machine consists of three linear tapes, and the operating device consists of three heads, each of which is the same as the head of a Turing machine and works with the corresponding tapes. Such machines are called *simple inductive Turing machines* (Burgin, 2005a).

The *control device* A is a finite automaton. It controls and regulates processes and parameters of the machine M : the state of the whole machine M , the processing of information by H , and the storage of information in the memory E .

The *memory* E of a general inductive Turing machines is divided into different but, as a rule, uniform cells. It is structured by a system of relations that organize memory as well-structured system and provide connections or ties between cells. In particular, *input* registers, the *working* memory, and *output* registers of M are discerned. Connections between cells form an additional structure K of E . Each cell can contain a symbol from an alphabet of the languages of the machine M or it can be empty.

In a general case, cells may be of different types. Different types of cells may be used for storing different kinds of data. For example, binary cells, which have type B , store bits of information represented by symbols 1 and 0. Byte cells (type BT) store information represented by strings of eight binary digits. Symbol cells (type SB) store symbols of the alphabet(s) of the machine M . Cells in conventional Turing machines have SB type. Natural number cells, which have type NN , are used in random access machines. Cells in the memory of quantum computers (type QB) store q -bits or quantum bits. Cells of the tape(s) of real-number Turing machines (Burgin, 2005a) have type RN and store real numbers. When different kinds of devices are combined into one, this new complex device may have several types of memory cells. In addition, different types of cells facilitate modeling the brain neuron structure by inductive Turing machines.

The *processor* H performs information processing in M . However, in comparison to computers, H performs very simple operations. When H consists of one unit, it can change a symbol in the cell that is observed by H , and go from this cell to another using a connection from K . It is possible that the processor H consists of several processing units similar to heads of a multihead Turing machine. This allows one to model various real and abstract computing systems: multiprocessor computers; Turing machines with several tapes; networks, grids and clusters of computers; cellular automata; neural networks; and systolic arrays.

The *software* R of the inductive Turing machine M is also a program that consists of simple rules:

$$q_h a_i \rightarrow a_j q_k c \quad (2.1)$$

Here q_h and q_k are states of A , a_i and a_j are symbols of the alphabet of M , and c is a type of connection in the memory E . The rule (2.1) means that if the state of the control device A of M is q_h and the processor H observes in the cell the symbol a_j , then the state of A becomes q_k , while the processor H writes the symbol a_j in the cell where it is situated and moves to the next cell by a connection of the type c . Each rule directs one step of computation of the inductive Turing machine M . Rules of the inductive Turing machine M define the transition function of M and describe changes of A , H , and E . Consequently, these rules also determine the transition functions of A , H , and E .

These rules cover only synchronous parallelism of computation. However, it is possible to consider inductive Turing machines of any order in which their processor can perform computations in the concurrent mode. Besides, it is also possible to consider inductive Turing machines with several processors. These models of computation are studied elsewhere.

A general step of the machine M has the following form. At the beginning, the processor H observes some cell with a symbol a_i (it may be Λ as the symbol of an empty cell) and the control device A is in some state q_h . Then the control device A (and/or the processor H) chooses from the system R of rules a rule r with the left part equal to $q_h a_i$ and performs the operation prescribed by this rule. If there is no rule in R with such a left part, the machine M stops functioning. If there are several rules with the same left part, M works as a nondeterministic Turing machine, performing all possible operations. When A comes to one of the final states from F , the machine M also stops functioning. In all other cases, it continues operation without stopping.

In the output stabilizing mode, M gives the result when M halts and its control device A is in a final state from F , or when M never stops but at some step of the computation the content of the output register becomes fixed and does not change (cf. Definition 3.4). The computed result of M is the word that is written in the output register of M . In all other cases, M does not give the result (cf. Definition 3.6).

Now let us build a constructive hierarchy of inductive Turing machines.

The memory E is called *recursive* if all relations that define its structure are recursive. Here recursive means that there are Turing machines that decide or build the structured memory (Burgin, 2005a). There are different techniques to organize this process. The simplest approach assumes that given some data, e.g., a description of the structure of E , a Turing machine T builds all connections in the memory E before the machine M starts its computation. According to another methodology, memory construction by the machine T and computations of the machine M go concurrently, while the machine M computes, the machine T constructs connections in the memory E . It is also possible to consider a situation when some connections in the memory E are assembled before the machine M starts its computation, while other connections are formed parallel to the computing process of the machine M .

Besides, it is possible to consider a schema when the machine T is separate from the machine M , while another construction adopts the machine T as a part of the machine M .

Inductive Turing machines with recursive memory are called *inductive Turing machines of the first order*.

While in inductive Turing machines of the first order, the memory is constructed by Turing machines or other recursive algorithms, it is possible to use inductive Turing machines for memory construction for other inductive Turing machines. This brings us to the concept of inductive Turing

machines of higher orders. For instance, in inductive Turing machines of the second order, the memory is constructed by Turing machines of the first order.

In general, we have the following definitions.

The memory E is called n -inductive if its structure is constructed by an inductive Turing machine of the order n . Inductive Turing machines with n -inductive memory are called *inductive Turing machines of the order $n + 1$* . Namely, in inductive Turing machines of order n , the memory is constructed by Turing machines of order $n - 1$.

We denote the class of all inductive Turing machines of the order n by \mathbf{IT}_n and take $\mathbf{IT} = \bigcup_{n=1}^{\infty} \mathbf{IT}_n$.

In such a way, we build a constructive hierarchy of inductive Turing machines. Algorithmic problems solved by these machines form a superrecursive hierarchy of algorithmic problems (Burgin, 2005b).

A simple inductive Turing machine has the same structure and the same rules (instructions) as a conventional Turing machine with three heads and three linear tapes: the input tape, output tape and working tape. The input tape is a read-only tape and the output tape is a write-only tape.

Computation or acceptance of a simple inductive Turing machine M consists of two stages. At first, M rewrites the input word from the input tape to the working tape. Then M starts working with this word in the working tape, writing something to the output tape from time to time.

Thus, the rules of a simple inductive Turing machine have the form

$$q_h(a_{i1}, a_{i2}, a_{i3}) \rightarrow (a_{j1}, a_{j2}, a_{j3})q_k(T_1, T_2, T_3) \quad (2.2)$$

The meaning of symbols in formula (2.2) is similar to notations used for Turing machines. Namely, we have:

- q_h and q_k are states of the control device A ;
- $a_{i1}, a_{i2}, a_{i3}, a_{j1}, a_{j2}$ and a_{j3} are symbols from the alphabet of M ;
- each of the symbols T_1, T_2 and T_3 is equal either to L , which denotes the transition of the head to the left adjacent cell or to R , which denotes the transition of the head to the right adjacent cell, or to N , which denotes absence of a head transition.

The rule (2.2) means that if the state of the control device A of M is q_h and the head h_t observes in the cell of the tape t the symbol a_{it} , then the state of A becomes q_k , while the head h_t writes the symbol a_j in the cell where it is situated and moves to the direction indicated by T_t ($t = 1, 2, 3$). Each rule directs one step of computation of the inductive Turing machine M .

It means that moves of a simple inductive Turing machine are the same as moves of a Turing machine with three tapes. The difference is in output. A Turing machine produces a result only when it halts. The result is a word on the output tape. A simple inductive Turing machine is also doing this but in addition, it produces its results without stopping. It is possible that in the sequence of computations after some step, the word on the output tape is not changing, while the simple inductive Turing machine continues working. This word, which is not changing, is the result of the machine that works in the output stabilizing computing mode. Thus, the simple

inductive Turing machine does not halt, producing a result after a finite number of computing operations.

Because here we consider inductive Turing machines that work only with finite words and processing of the input word starts only after it is rewritten into the working tape, it is possible to assume that there is no input tape, the initial word is written in the working tape and the rules have the form

$$q_h(a_{i_2}, a_{i_3}) \rightarrow (a_{j_2}, a_{j_3})q_k(T_2, T_3) \quad (2.3)$$

Here a_{i_2} and a_{j_2} are symbols in the working tape, a_{i_3} and a_{j_3} are symbols in the output tape, the symbol T_2 directs the move of the head h_2 , while the symbol T_3 directs the move of the head h_3 .

Besides, it is also possible to assume that the output written in the output tape does not influence operations of the inductive Turing machine because the output tape is the write-only tape.

3. Comparing results of stabilizing computations

In this section, we study functioning of inductive Turing machines of the first order with one linearly ordered output register and one linearly ordered input register (Burgin, 2005a). We also assume that these inductive Turing machines work with finite words in some alphabet. The main emphasis here is on simple inductive Turing machines. Note that in general, inductive Turing machines can work not only with finite and infinite words but also with multidimensional arrays, graphs and even more sophisticated data structures.

In the previous section, we considered only the output stabilizing computing mode of inductive Turing machines. However, it is possible to use other modes of inductive Turing machine functioning, for example, the state stabilizing mode. This is similar to the functioning modes of finite automata that work with infinite words, (Burks & Wright, 1953; Büchi, 1960; Thomas, 1990; Chadha *et al.*, 2009).

The simplest modes of inductive Turing machine functioning are acceptance by halting and computation by halting. Namely, we have:

Definition 3.1. a) An inductive Turing machine M accepts the input word *by halting* if after some number of steps, the machine M stops.

b) The set $L_{ht}(M)$ of all words accepted by halting of an inductive Turing machine M is called the *halting accepted language* of the machine M .

This is the standard mode for Turing machines (Hopcroft *et al.*, 2001; Burgin, 2005a).

Computation by halting is defined in a similar way.

Definition 3.2. a) An inductive Turing machine M computes a word w *by halting* if after some number of steps, the machine M stops and when this happens, w is the word in output tape.

b) The set $L^{ht}(M)$ of all words computed by halting of an inductive Turing machine M is called the *halting computed language* of the machine M .

In the theory of inductive Turing machines, it is proved that when an inductive Turing machine M gives the result by halting, i.e., accepts a word by halting or computes a word by halting, M is linguistically equivalent to a Turing machine, i.e., the language produced (accepted or computed) by M can be produced by some Turing machine (Burgin, 2005a). In the theory of Turing machines, it is proved that acceptance by halting is linguistically equivalent to computation by halting. This gives us the following result.

Proposition 3.1. *Computation by halting is linguistically equivalent to acceptance by halting in the class of all inductive Turing machines of the first order.*

Thus, inductive Turing machine M of the first order can compute and accept all recursively enumerable languages and only these languages.

Now we will study more productive modes of inductive Turing machine functioning when machines are able to compute or accept languages that are not recursively enumerable.

Let us consider an inductive Turing machine M . In the set Q of states of the machine M , several subsets F_1, \dots, F_k are selected and called the *final groups of states* of the inductive Turing machine M .

Definition 3.3. An inductive Turing machine M gives a *result by state stabilizing* if after some number of steps, the state of the machine M always remains in the same final group of states.

Note that traditionally states of the control device of a Turing machine or of an inductive Turing machine are treated as states of the whole machine (Burgin, 2005a; Hopcroft *et al.*, 2001; Sipser, 1996). We follow this tradition.

It is possible that one or several final groups consist of a single state. Then stabilization in such a group means that the state of the machine M always stops changing after some number of steps.

When the processing mode is acceptance, the result is acceptance of the input word. We formalize this situation by the following definition.

Definition 3.4. a) An inductive Turing machine M accepts the input word *by state stabilizing* if after some number of steps, the state of the machine M always remains in the same final group of states.

b) The set $L_{st}(M)$ of all words accepted by state stabilizing of an inductive Turing machine M is called the *state stabilizing accepted language* of the machine M .

It is natural to consider the state stabilizing language $L_{st}(M)$ of the machine M as the result of M working in the acceptance mode.

We denote by $\mathbf{L}_{st}(\text{ITM1})$ the set of all state stabilizing accepted languages of inductive Turing machines of the first order and by $\mathbf{L}_{st}(\text{SITM})$ the set of all state stabilizing accepted languages of simple inductive Turing machines.

In the computing mode, the result is defined in a different way, which is formalized by the following definition.

Definition 3.5. a) An inductive Turing machine M computes the input word w by *state stabilizing* if after some number of steps, the state of the machine M always remains in the same final group of states and w is the first word in the output tape when the state stabilization process starts. This output w is the final result of the machine M .

b) The set $L^{st}(M)$ of all words computed by state stabilizing of an inductive Turing machine M is called the *state stabilizing computed language* of the machine M .

It is natural to consider the state stabilizing language $L_{or}(M)$ of the machine M as the result of M working in the acceptance mode.

We denote by $\mathbf{L}^{st}(\text{ITM1})$ the set of all state stabilizing computed languages of inductive Turing machines of the first order and by $\mathbf{L}^{st}(\text{SITM})$ the set of all state stabilizing computed languages of simple inductive Turing machines.

Definition 3.6. An inductive Turing machine M gives a *result by output stabilizing* if after some number of steps the output of the machine M stops changing. This output is the final result of the machine M .

When the processing mode is acceptance, the result is acceptance of the input word. Namely, we have the following concept.

Definition 3.7. a) An inductive Turing machine M accepts the input word by *output stabilizing* if after some number of steps, the output of the machine M stops changing. This output is the final result of the machine M .

b) The set $L_{or}(M)$ of all words accepted by output stabilizing of an inductive Turing machine M is called the *output stabilizing accepted language* of the machine M .

It is natural to consider the output stabilizing language $L_{or}(M)$ of the machine M as the result of M working in the acceptance mode.

We denote by $\mathbf{L}_{or}(\text{ITM1})$ the set of all output stabilizing accepted languages of inductive Turing machines of the first order and by $\mathbf{L}_{or}(\text{SITM})$ the set of all state stabilizing accepted languages of simple inductive Turing machines.

Definition 3.8. a) An inductive Turing machine M computes the input word by *output stabilizing* if after some number of steps, the output of the machine M stops changing. This output is the final result of the machine M .

b) The set $L^{ot}(M)$ of all words computed by output stabilizing of an inductive Turing machine M is called the *output stabilizing computed language* of the machine M .

It is natural to consider the state stabilizing language $L^{ot}(M)$ of the machine M as the result of M working in the computation mode.

We denote by $\mathbf{L}^{ot}(\text{ITM1})$ the set of all state stabilizing computed languages of inductive Turing machines of the first order and by $\mathbf{L}^{ot}(\text{SITM})$ the set of all state stabilizing computed languages of simple inductive Turing machines.

Let us consider more restrictive modes of inductive Turing machine functioning.

Definition 3.9. An inductive Turing machine M gives a *result by bistabilizing* if after some number of steps the output of the machine M stops changing, while the state of M remains in the same final group of states.

This output is the final result of the machine M .

When the processing mode is acceptance, the result is acceptance of the input word. Namely, we have the following concept.

Definition 3.10. a) An inductive Turing machine M accepts the input word *by bistabilizing* if after some number of steps, the output of the machine M stops changing, while the state of M remains in the same final group of states.

b) The set $L_{bt}(M)$ of all words accepted by bistabilizing of an inductive Turing machine M is called the *bistabilizing accepted language* of the machine M .

It is natural to consider the output stabilizing language $L_{bt}(M)$ of the machine M as the result of M working in the acceptance mode.

We denote by $\mathbf{L}_{bt}(\text{ITM1})$ the set of all bistabilizing accepted languages of inductive Turing machines of the first order and by $\mathbf{L}_{bt}(\text{SITM})$ the set of all bistabilizing accepted languages of simple inductive Turing machines.

Definition 3.11. a) An inductive Turing machine M computes the input word *by bistabilizing* if after some number of steps, the output of the machine M stops changing, while the state of M remains in the same final group of states.

The output that stopped changing is the final result of the machine M .

b) The set $L^{bt}(M)$ of all words computed by bistabilizing of an inductive Turing machine M is called the *bistabilizing computed language* of the machine M .

It is natural to consider the bistabilizing language $L^{bt}(M)$ of the machine M as the result of M working in the computation mode.

We denote by $\mathbf{L}^{bt}(\text{ITM1})$ the set of all bistabilizing computed languages of inductive Turing machines of the first order and by $\mathbf{L}^{bt}(\text{SITM})$ the set of all bistabilizing computed languages of simple inductive Turing machines.

Definitions imply the following results.

Proposition 3.2. For any inductive Turing machine M , we have:

a) $L^{bt}(M) \subseteq L^{ot}(M)$.

b) $L^{bt}(M) \subseteq L^{st}(M)$.

c) $L_{bt}(M) \subseteq L_{ot}(M)$.

d) $L_{bt}(M) \subseteq L_{st}(M)$.

Corollary 3.1. The following inclusions are true:

- a) $\mathbf{L}^{bt}(\text{ITM1}) \subseteq \mathbf{L}^{ot}(\text{ITM1})$.
- b) $\mathbf{L}_{bt}(\text{ITM1}) \subseteq \mathbf{L}_{ot}(\text{ITM1})$.
- c) $\mathbf{L}_{bt}(\text{ITM1}) \subseteq \mathbf{L}_{st}(\text{ITM1})$.
- d) $\mathbf{L}^{bt}(\text{ITM1}) \subseteq \mathbf{L}^{st}(\text{ITM1})$.
- e) $\mathbf{L}^{bt}(\text{SITM}) \subseteq \mathbf{L}^{ot}(\text{SITM})$.
- f) $\mathbf{L}_{bt}(\text{SITM}) \subseteq \mathbf{L}_{ot}(\text{SITM})$.
- g) $\mathbf{L}_{bt}(\text{SITM}) \subseteq \mathbf{L}_{st}(\text{SITM})$.
- h) $\mathbf{L}^{bt}(\text{SITM}) \subseteq \mathbf{L}^{st}(\text{SITM})$.

Computation by output stabilizing is the basic mode of inductive Turing machines when they perform computations (Burgin, 2005a). Properties of inductive Turing machines show that an inductive Turing machine M of the first order that accepts (or computes) by halting is linguistically equivalent to an accepting (or computing) Turing machine (Burgin, 2005a). At the same time, in general inductive Turing machines of the first order are essentially more powerful than Turing machines. For instance, there are simple inductive Turing machines that solve the halting problem for all Turing machines. This gives us the following result.

Proposition 3.3. *For any inductive Turing machine M , we have:*

- a) *Computation by state stabilizing is not linguistically equivalent to computation by halting in the class of all inductive Turing machines of the first order.*
- b) *Acceptation by state stabilizing is not linguistically equivalent to acceptation by halting in the class of all inductive Turing machines of the first order.*

Note that halting is a very specific case of stabilizing in which the process simply stops. However, it is more natural to compare non-stopping processes of acceptance and computation for inductive Turing machines.

Comparing the state stabilizing computed language $L^{st}(M)$ of an inductive Turing machine M and the output stabilizing computed language $L^{ot}(M)$ of the machine M , we see that in general these languages do not coincide. The same can be true for the accepted languages of the inductive Turing machine M . Such machines are considered in the following examples.

Example 3.1. Let us take a simple inductive Turing machine M such that works in the alphabet $\{0, 1\}$ and given a word w as its input, changes w to the word $w1$, i.e., M writes 1 at the end of w , gives as the output and repeats this operation with the output without stopping. As the output of M is changing on each step, the output stabilizing accepted language $L_{ot}(M)$ of the machine M is empty. At the same time, if we take all states of M as a single final group, then the state stabilizing accepted language $L_{st}(M)$ of the machine M contains all words in the alphabet $\{0, 1\}$. Thus, $L_{ot}(M) \neq L_{st}(M)$.

The same inequality is true for the computed language of the inductive Turing machine M . Indeed, the output stabilizing computed language $L^{ot}(M)$ of the machine M is empty because its output never stops changing. At the same time, the machine M computes by the state stabilizing all words in the alphabet $\{0, 1\}$ that has 1 at the end. Thus, $L^{ot}(M) \neq L^{st}(M)$.

In Example 3.1, as we can see, the state stabilizing accepted language $L_{st}(M)$ of the machine M is much larger than the output stabilizing accepted language $L_{ot}(M)$ of the same machine M . The same inequality is true for the computed languages of inductive Turing machine M . However, the opposite situation is also possible.

Example 3.2. Let us take a simple inductive Turing machine W such that works in the alphabet $\{0, 1\}$ and given a word w as its input, gives this word w as the output and repeats this operation with the output and then never produces any new output. At the same time, it is possible to program the machine W so that after it rewrites w into its output tape, the machine W goes into an infinite cycle changing the state on each step. As the result, the output stabilizing accepted language $L_{ot}(W)$ of the machine W contains all words in the alphabet $\{0, 1\}$. However, if we do not define any final group in the states of the machine W , then the state stabilizing accepted language $L_{st}(W)$ of the machine W is empty. Thus, $L_{ot}(W) \neq L_{st}(W)$.

The same inequality is true for the computed languages of inductive Turing machine W . Indeed, the state stabilizing computed language $L^{ot}(W)$ of the machine W is empty because its state never stops changing and there are no final state groups. At the same time, the machine W computes by the output stabilizing all words in the alphabet $\{0, 1\}$ that has 1 at the end. Thus, $L^{ot}(W) \neq L^{st}(W)$.

Thus, the output stabilizing accepted language $L_{ot}(W)$ of the machine W is much larger than the state stabilizing accepted language $L_{st}(W)$ of the same machine W . The same inequality is true for the computed languages of inductive Turing machine W .

However, for the classes of the output stabilizing accepted by inductive Turing machines of the first order languages and the state stabilizing accepted by inductive Turing machines of the first order languages this is not true.

Theorem 3.1. *If a language L has an inductive Turing machine of the first order that computes L by output stabilizing, then L has an inductive Turing machine of the first order that computes L by bistabilizing.*

Proof. Let us consider an inductive Turing machine M of the first order that computes a language L by output stabilizing and construct an inductive Turing machine of the first order that computes a language L by bistabilizing. In the theory of inductive Turing machines, it is proved that any an inductive Turing machine of the first order is equivalent to a simple inductive Turing machine M that never stops given some input (Burgin, 2005a). Thus, it is possible to assume that M is a simple inductive Turing machine that never stops given some input and accepts by output stabilizing. Note that in the mode of output stabilizing final groups of states do not play any role.

Thus, it is possible to use the same machine M for bistabilizing computation of L . Indeed, making the set Q of all states of M as one final group, we see that according to Definition 11, the machine M computes the same language $L(M)$ as before by bistabilizing because the state is always stabilized. Theorem is proved. \square

Results from (Burgin, 2005a) show that $L_{st}(\text{ITM1}) = L_{st}(\text{SITM})$ and $L_{ot}(\text{ITM1}) = L_{ot}(\text{SITM})$. Thus, Theorem 3.1 implies the following result.

Corollary 3.2. $L^{ot}(\text{ITM1}) \subseteq L^{bt}(\text{ITM1})$.

This result shows that computation by bistabilizing looks more powerful than computation by output stabilizing in the class of all inductive Turing machines of the first order. However, the inverse of Theorem 3.1 is also true.

Theorem 3.2. *If a language L has an inductive Turing machine M of the first order that computes L by bistabilizing, then L has an inductive Turing machine K of the first order that computes L by output stabilizing.*

Proof. Let us consider an inductive Turing machine M of the first order that computes a language L by bistabilizing and construct an inductive Turing machine K of the first order that computes a language L by output stabilizing. As before, it is possible to assume that M is a simple inductive Turing machine.

To allow functioning in the bistabilizing mode, in the set Q of states of M , several subsets F_1, \dots, F_k are selected as final groups of states of the inductive Turing machine M .

By Proposition 3.2, $L = L^{bt}(M) \subseteq L^{ot}(M)$, i.e., the language $L^{ot}(M)$ computable by output stabilizing may have words that do not belong to the language $L^{bt}(M)$ computable by bistabilizing. To prove the necessary result, we show how to get rid of these extra words by appropriately changing the machine M .

Such an extra word w is computed by M when the output stabilizes but the state does not remain in the same final group. To prevent this, we add new symbols $b_1, b_2, b_3, \dots, b_m$ to the alphabet $A = \{a_1, a_2, a_3, \dots, a_m\}$ of the machine M . Then we consider all instructions of the form

$$q_h(a_{i_2}, a_{i_3}) \rightarrow (a_{j_2}, a_{i_3})q_k(T_2, T_3)$$

where q_h belongs to some final group of states F_t , while q_k does not belong to this group and change this instruction to the two following instructions

$$q_h(a_{i_2}, a_{i_3}) \rightarrow (a_{j_2}, b_{i_3})q_k(T_2, T_3)$$

$$q_k(a_{i_2}, b_{i_3}) \rightarrow (a_{j_2}, a_{i_3})q_k(T_2, T_3)$$

We do not change other instructions of M and in such a way, we obtain a simple inductive Turing machine K . As a result of these changes, the output of K always changes when the state leaves some final group. So, the output stabilizes if and only if the state stabilizes in some final group. Consequently, the new inductive Turing machine K computes the given language L . Theorem is proved. \square

Theorem 3.2 implies the following result.

Corollary 3.3. $L^{bt}(\text{ITM1}) \subseteq L^{ot}(\text{ITM1})$.

This result shows that computation by output stabilizing looks more powerful than computation by bistabilizing in the class of all inductive Turing machines of the first order. However, Theorems 3.1 and 3.2 together imply that these modes are linguistically equivalent.

Theorem 3.3. *Computation by output stabilizing is linguistically equivalent to computation by bistabilizing in the class of all inductive Turing machines of the first order.*

Corollary 3.4. $L^{bt}(\text{ITM1}) = L^{ot}(\text{ITM1})$.

Let us consider relations between computed and accepted languages.

Theorem 3.4. *If a language L has an inductive Turing machine of the first order that computes L by state stabilizing, then L has an inductive Turing machine of the first order that computes L by bistabilizing.*

Proof. Let us consider an inductive Turing machine M of the first order that computes a language L by state stabilizing and construct an inductive Turing machine K of the first order that computes a language L by bistabilizing. As before, it is possible to assume that M is a simple inductive Turing machine.

To allow functioning in the state stabilizing mode, in the set Q of states of M , several subsets F_1, \dots, F_k are selected as final groups of states of the inductive Turing machine M .

We begin our construction of the machine K by adding one more working tape to the tapes the machine M . A simple inductive Turing machine has only three tapes (see Section 2). However, by the standard technique described, for example, in (Hopcroft *et al.*, 2001) or in (Sipser, 1996), it is possible to show that we can build a simple inductive Turing machine which simulates two working tapes using only one working tape and accepting the same language. Thus, adding one more working tape, we do not extend the class of accepted languages.

As a result of this action, the machine K has an input tape T_{in} , an output tape T_{out} and two working tapes T_{1w} and T_{2w} . We use the tape T_{1w} for exact modeling of the working tape T_w of the machine M . To do this, we preserve all parts of the rules that are related to the tape T_w in the machine M .

At the same time, we use the tape T_{2w} for exact modeling of the output tape T_{Mout} of the machine M . To do this, we redirect all parts of the rules that are related to the tape T_{Mout} in the machine M , making them the rules for the tape T_{2w} in K .

Besides, we add the rewriting state r to the set of states of the machine M and new rules for the tape T_{out} in K . The rules in which change of the state goes in one and the same final group are preserved in K . This allows the following operations. When the state of M leaves a final group, the same happens with the machine K according to its rules. While the machine M continues its functioning, comes to the rewriting state r , erases everything from the tape T_{out} in K and then rewrites the word from the tape T_{2w} into the output tape T_{out} . When the state of M is outside any final group and changes, the machine K repeats the same steps.

Thus, the output of K always changes when the state leaves some final group or is outside any final group and changes. As a result, the output of K stabilizes if and only if the state remains inside some final group. Besides, it stabilizes on the first word that was on the output tape of M when the stabilization of states started. So, the machine K computes the language L by bistabilizing. Theorem is proved. \square

Theorem 3.4 implies the following result.

Corollary 3.5. $L^{st}(\text{ITM1}) \subseteq L^{bt}(\text{ITM1})$.

Note that the machine K constructed in the proof of Theorem 3.4 also computes the language L by output stabilizing. This gives us the following result.

Theorem 3.5. *If a language L has an inductive Turing machine of the first order that computes L by state stabilizing, then L has an inductive Turing machine of the first order that computes L by output stabilizing.*

Theorem 3.5 implies the following result.

Corollary 3.6. $L^{st}(\text{ITM1}) \subseteq L^{ot}(\text{ITM1})$.

Inversion of Theorem 3.4 is also true.

Theorem 3.6. *If a language L has an inductive Turing machine of the first order that computes L by bistabilizing, then L has an inductive Turing machine of the first order that computes L by state stabilizing.*

Proof. Let us consider an inductive Turing machine M of the first order that computes a language L by bistabilizing and construct an inductive Turing machine K of the first order that computes a language L by state stabilizing. As before, it is possible to assume that M is a simple inductive Turing machine.

To allow functioning in the bistabilizing mode, in the set $Q = \{q_1, q_2, q_3, \dots, q_m\}$ of states of M , several subsets F_1, \dots, F_k are selected as final groups of states of the inductive Turing machine M .

By Proposition 3.2, $L = L^{bt}(M) \subseteq L^{st}(M)$, i.e., the language $L^{st}(M)$ computable by output stabilizing may have words that do not belong to the language $L^{bt}(M)$ computable by bistabilizing. To prove the necessary result, we show how to get rid of these extra words by appropriately changing the machine M .

Such an extra word w is computed by M when the state stabilizes in some final group but the output does not remain the same all the time. To prevent this, we add a new states $p_1, p_2, p_3, \dots, p_m$ to the set Q of states of the machine M without changing the final groups. Then we consider all instructions of the form.

$$q_h(a_{i_2}, a_{i_3}) \rightarrow (a_{j_2}, a_{j_3})q_k(T_2, T_3)$$

where q_h and q_k belong to some final group of states F_n but $a_{i_3} \neq a_{j_3}$ and change this instruction to the two following instructions

$$q_h(a_{i_2}, a_{i_3}) \rightarrow (a_{j_2}, a_{j_3})p_k(T_2, T_3)$$

$$p_k(a_{i_2}, a_{i_3}) \rightarrow (a_{j_2}, a_{j_3})q_k(T_2, T_3)$$

We do not change other instructions of M and in such a way, we obtain a simple inductive Turing machine K . As a result of these changes, if the state belongs to a final group, it always leaves this group when the output of K changes. So, the state stabilizes in some final group if and only if the output stabilizes. Consequently, the new inductive Turing machine K computes the given language L by state stabilizing. Theorem is proved. \square

Theorem 3.6 implies the following result.

Corollary 3.7. $L^{bt}(\text{ITM1}) \subseteq L^{st}(\text{ITM1})$.

This result shows that computation by output stabilizing looks more powerful than computation by bistabilizing in the class of all inductive Turing machines of the first order. However, Theorems 3.4 and 3.6 together imply that these modes are linguistically equivalent.

Theorem 3.7. *Computation by state stabilizing is linguistically equivalent to computation by bistabilizing in the class of all inductive Turing machines of the first order.*

Corollary 3.8. $L^{bt}(\text{ITM1}) = L^{st}(\text{ITM1})$.

Theorems 3.3 and 3.7 together imply that all considered computing modes are linguistically equivalent.

Theorem 3.8. *Computation by state stabilizing, computation by output stabilizing and computation by bistabilizing are linguistically equivalent in the class of all inductive Turing machines of the first order.*

Corollary 3.9. $L^{bt}(\text{ITM1}) = L^{st}(\text{ITM1}) = L^{ot}(\text{ITM1})$.

Inductive Turing machines allow modeling Turing machines, namely, for any Turing machine T , there is an inductive Turing machine M that has the same language as P (Burgin, 2005a). This makes possible to obtain the classical result of computability theory that computing by final state is linguistically equivalent to computing by halting (Burgin, 2005a; Hopcroft *et al.*, 2001; Sipser, 1996) as a direct corollary of Theorem 3.8.

Comparing the state stabilizing computed language $L^{st}(M)$ of an inductive Turing machine M , the output stabilizing computed language $L^{ot}(M)$ of the machine M and the bistabilizing computed language $L^{bt}(M)$ of the machine M , we see that in general these languages do not coincide. The same can be true for the accepted languages of the inductive Turing machine M . Such machines are considered in the following examples.

Indeed, the output stabilizing accepted language $L_{ot}(W)$ of the machine W from Example 3.2 is much larger than the bistabilizing accepted language $L_{bt}(W)$ of the same machine W . The same inequality is true for the computed languages of inductive Turing machine W . In addition, the state stabilizing accepted language $L_{st}(M)$ of the machine M is much larger than the bistabilizing accepted language $L_{bt}(M)$ of the same machine M . The same inequality is true for the computed languages of inductive Turing machine M .

However, Proposition 3.2 shows that for any Turing machine M , the language $L_{bt}(M)$ cannot be larger than the language $L_{st}(M)$ and the language $L_{bt}(M)$ cannot be larger than the language $L_{ot}(M)$.

In addition, for the classes of the output stabilizing accepted by inductive Turing machines of the first order languages and the state stabilizing accepted by inductive Turing machines of the first order languages this is not true.

Now let us explore for inductive Turing machines of the first order, relations between computed and accepted languages and classes of languages.

Theorem 3.9. *If a language L has an inductive Turing machine M of the first order that accepts L by state stabilizing, then L has an inductive Turing machine V of the first order that computes L by state stabilizing.*

Proof. Let us consider an inductive Turing machine M of the first order that accepts a language L by state stabilizing, i.e., $L = L_{st}(M)$. As it is proved that any an inductive Turing machine of the first order is equivalent to a simple inductive Turing machine M (Burgin, 2005a), it is possible to assume that M is a simple inductive Turing machine.

Then we change the rules R of the inductive Turing machine M to the rules P of the inductive Turing machine V by the following transformation. We exclude from all rules of the machine M any possibility to write something into the output tape. Besides, we add such rules according to which the process of functioning machine V begins so that the output head writes the input word into the output tape.

By construction, V is also a simple inductive Turing machine.

As there no other transformations of the system of initial rules, the output of V is never changing. By Definition 3.3, this output is the result of computation for V if and only if after some number of steps, the state of the machine M always remains in the same final group of states. It means that a word w is accepted by state stabilizing in the inductive Turing machine M if and only if the word w is computed by state stabilizing in the inductive Turing machine V . Consequently, languages $L_{st}(M)$ and $L^{st}(V)$ coincide. Theorem is proved. \square

Corollary 3.10. $L_{st}(\text{ITM1}) \subseteq L^{st}(\text{ITM1})$.

This result shows that computation by state stabilizing looks more powerful than acceptance by state stabilizing in the class of all inductive Turing machines of the first order.

Theorem 3.10. *If a language L has an inductive Turing machine M of the first order that computes L by output stabilizing, then there is an inductive Turing machine W of the first order that accepts L by output stabilizing.*

Proof. Let us consider an inductive Turing machine M of the first order that computes a language L by output stabilizing, i.e., $L = L^{ot}(M)$. As it is proved that any an inductive Turing machine of the first order is equivalent to a simple inductive Turing machine M (Burgin, 2005a), it is possible to assume that M is a simple inductive Turing machine.

In addition, we consider a Turing machine G that given a word 1^n , generates n different words in the alphabet X of the machine M , generating all words in the alphabet X in such a way. We also take a Turing machine C that compares its input with the word written in the tape of this machine and called the sample word of C . When both words are equal, the machine C gives 1 as its output and halts. Otherwise, the machine C gives 0 as its output and halts.

This allows us to build the machine W in the following way. It contains subroutines G_0 , C_0 and M_0 , that simulate the machines G , C and M , respectively. The machine W has as many working tapes as it is necessary for functioning of the subroutines G_0 , C_0 and M_0 . In particular, two counting tapes are added - the counting tape for the machine W and the counting tape for the subroutine M_0 . In these tapes, numbers of iterations are stored. Then we add rules for W such that allow it to perform the following steps.

1. When a word w comes to W as its input, the machine W writes w into the working tape of the machine C as its sample word and goes to the step 2.
2. The machine W writes the number 1 into the counter tape, which contains the number of iterations, and goes to the step 3.
3. The machine W gives the number 1 as the input to the subroutine G_0 and goes to the step 4.
4. The subroutine G_0 generates the word u_1 and gives it as the input to the subroutine M_0 , going to the step 5.
5. The subroutine M_0 computes with this input until the first word w_1 appears in the output tape of M_0 . Then the machine W goes to the step 6.
6. The machine W writes the word w_1 in its output tape and gives w_1 as the input to the subroutine C_0 , which compares w_1 and w . Then the machine W goes to the step 7 or 9 depending on the output of C_0 .
7. If the output of C_0 is 1, then the subroutine M_0 continues its computations until the next output word, in this case w_2 , appears in the output tape of M_0 . Then the machine W goes to the step 8.
8. The machine W gives w_2 as the input to the subroutine C_0 , which compares w_2 and w . Then the machine W goes to the step 7 or 9 depending on the output of C_0 . Note that if the output of C_0 is always 1 starting from some input, the machine W accepts w by output stabilizing.
9. If the output of C_0 is 0, then the machine W writes the word w in its output tape, changes this word to the word checked by C_0 (it will be w_1 if the previous step had number 6, while it will be w_2 if the previous step had number 8) and goes to the step 10.
10. The machine W adds 1 to the number in its counter tape, gives this new number n (in the second iteration $n = 2$) as the input to the subroutine G_0 and goes to the step 11.
11. The subroutine G_0 generates n words u_1, u_2, \dots, u_n and gives all of them one by one as the inputs to the subroutine M_0 .
12. The subroutine M_0 writes 1 into its counting tape and computes with the input u_1 until it writes n words into the output tape. Then the machine W goes to the step 13.
13. The machine W gives the current output word w_k of the subroutine M_0 as the input to the subroutine C_0 , which compares w_k and w . Then the machine W goes to the step 14 or 15 depending on the output of C_0 .
14. If the output of C_0 is 1, then the subroutine M_0 continues its computations until the next output word, say w_r , appears in the output tape of M_0 . Note that if the output of C_0 is always 1 starting from some input, the machine W accepts w by output stabilizing.

15. If the output of C_0 is 0 and the number t in its counting tape is less than k , then the machine W writes the word w in its output tape and changes this word to the word checked by C_0 , while the subroutine M_0 adds 1 to the number t in its counting tape and computes with the input u_{t+1} until it writes n words into the output tape. Then the machine W goes to the step 13.
16. If the number t in its counting tape of M_0 becomes equal to k , the machine W goes to the step 10.

If the word w is computed by output stabilizing of the inductive Turing machine M , then given some input u , the machine M makes some number of steps, and then its output becomes equal to w and stops changing. In this case, the output of C_0 is always 1 starting after some number of steps, and consequently, the machine W accepts w by output stabilizing.

If the word w is not computed by output stabilizing of the inductive Turing machine M for any input, then either the output of M is not stabilizing or it is stabilizing with the word v that is not equal to w . In the first case, the output of W is also not stabilizing and thus, the machine W does not accept w by output stabilizing. In the second case, the output of W is also not stabilizing because both words w and v appear infinitely many times in the output tape of W and thus, the machine W does not accept w by output stabilizing.

By construction, W is a simple inductive Turing machine.

Thus, the simple inductive Turing machine W does not accept w by output stabilizing is and only if the simple inductive Turing machine M does not accept w by output stabilizing.

Theorem is proved because w is an arbitrary word in the alphabet of the inductive Turing machine M . □

Corollary 3.11. $L^{ot}(\text{ITM1}) \subseteq L_{ot}(\text{ITM1})$.

This result shows that acceptance by output stabilizing looks more powerful than computation by output stabilizing in the class of all inductive Turing machines of the first order.

Let us take an inductive Turing machine M functioning of which satisfies Condition ST and its state stabilizing language $L^{st}(M)$, i.e., the set of all words computed by state stabilizing of the machine M (cf. Definition 3.3). Thus, a word w is computed by state stabilizing of the machine M if and only if is computed by output stabilizing of the machine M . Consequently, $L^{st}(M) = L^{ot}(M)$. This gives us the following results.

Theorem 3.11. For any inductive Turing machine M of the first order functioning of which satisfies Condition ST, $L^{st}(M) \subseteq L^{ot}(M)$.

Corollary 3.12. $L^{st}(\text{ITM1}) \subseteq L^{ot}(\text{ITM1})$.

This result shows that computation by output stabilizing looks more powerful than computation by state stabilizing in the class of all inductive Turing machines of the first order.

Note that in general Theorem 11 does not imply equality of the classes $L^{st}(\text{ITM1})$ and $L^{ot}(\text{ITM1})$ because not all inductive Turing machines satisfy Condition ST.

At the same time, other results obtained in this paper allows us to find more exact relations between these classes of languages. Namely, by Corollary 3.1, we have $L_{st}(\text{ITM1}) \subseteq L_{ot}(\text{ITM1})$.

By Corollary 3.2, we have $\mathbf{L}^{ot}(\text{ITM1}) \subseteq \mathbf{L}^{bt}(\text{ITM1})$.

Comparing acceptance by state with acceptance by output, we obtain the following inclusion $\mathbf{L}_{ot}(\text{ITM1}) \subseteq \mathbf{L}_{st}(\text{ITM1})$.

By Corollary 3.4, we have $\mathbf{L}^{bt}(\text{ITM1}) \subseteq \mathbf{L}^{ot}(\text{ITM1})$.

By Corollary 3.10, we have $\mathbf{L}_{st}(\text{ITM1}) \subseteq \mathbf{L}^{st}(\text{ITM1})$.

By Corollary 3.11, we have $\mathbf{L}^{ot}(\text{ITM1}) \subseteq \mathbf{L}_{ot}(\text{ITM1})$.

By Corollary 3.12, we have $\mathbf{L}^{st}(\text{ITM1}) \subseteq \mathbf{L}^{ot}(\text{ITM1})$.

This gives us the following chain of inclusions:

$$\mathbf{L}^{st}(\text{ITM1}) \subseteq \mathbf{L}^{ot}(\text{ITM1}) \subseteq \mathbf{L}_{ot}(\text{ITM1}) \subseteq \mathbf{L}_{st}(\text{ITM1}) \subseteq \mathbf{L}^{st}(\text{ITM1}).$$

By properties of sets and the inclusion relation, the chain (5) implies that all inclusions in it are equalities. Thus, we have the following result.

Theorem 3.12. *In the class of all inductive Turing machines of the first order, the following modes of functioning are linguistically equivalent:*

1. *Acceptation by state stabilizing.*
2. *Acceptation by output stabilizing.*
3. *Computation by state stabilizing.*
4. *Computation by output stabilizing.*

Inductive Turing machines allow modeling Turing machines, namely, for any Turing machine T , there is an inductive Turing machine M that has the same language as P (Burgin, 2005a). This makes possible to obtain the following classical result of computability theory as a direct corollary of Theorem 3.12.

Proposition 3.4. *. For Turing machines, the acceptation mode is linguistically equivalent to the computation mode, i.e., the class of languages accepted by Turing machines coincides with the class of languages computed by Turing machines.*

This result justifies the situation when in the majority of textbooks on computer science, it is assumed that Turing machines work in the acceptation mode and model computers and this allows modeling computers although computers, as a rule, work in the computation mode.

4. Conclusion

Various models of computation are studied in computer science - deterministic and nondeterministic finite automata, deterministic and nondeterministic pushdown automata, deterministic and nondeterministic Turing machines with one or many tapes, which can be one-dimensional and many-dimensional, and so on.

The basic results in this area are theorems on linguistic equivalence of different models of computation, i.e., equivalence with respect to the languages that are accepted/generated by these models. Thus, for finite automata, it is proved that the class of languages accepted by deterministic finite automata is the same as the class of languages accepted by nondeterministic finite automata

and as the class of languages accepted by nondeterministic finite automata with ε -transitions (cf., for example, (Hopcroft *et al.*, 2001): Theorem 2.12, Theorem 2.22; (Sipser, 1996): Theorem 1.19)).

In the theory of pushdown automata, it is proved that the class of languages accepted by pushdown automata by final state is the same as the class of languages accepted by pushdown automata by empty stack and as the class of languages generated by context free grammars (cf., for example, (Hopcroft *et al.*, 2001): Theorem 6.9, Theorem 6.11, Theorem 6.14; (Sipser, 1996): Theorem 2.12).

In the theory of Turing machines, it is proved that the class of languages accepted by deterministic Turing machines with a single tape is the same as the class of languages accepted by nondeterministic Turing machines with a single and as the class of languages accepted by Turing machines with many tapes and as the class of languages accepted by Turing machines with multidimensional tapes and as the class of languages accepted by pushdown automata with two and more stacks (cf., for example, (Hopcroft *et al.*, 2001): Theorem 8.9, Theorem 8.11, Theorem 6.14, Theorem 8.13; (Sipser, 1996): Theorem 3.8, Theorem 3.10).

In this paper, we obtained similar results for inductive Turing machines. Namely, it is proved that: (1) the class of languages computed by output stabilizing of inductive Turing machines of the first order is the same as the class of languages computed by bistabilizing of inductive Turing machines of the first order (Theorem 3.3); (2) the class of languages computed by output stabilizing of inductive Turing machines of the first order is the same as the class of languages computed by state stabilizing of inductive Turing machines of the first order (Theorem 3.8); (2) the class of languages computed by output (state) stabilizing of inductive Turing machines of the first order is the same as the class of languages accepted by output (state) stabilizing of inductive Turing machines of the first order (Theorem 3.12).

It is necessary to remark that it is possible to include the results of this paper into a standard course of the theory of automata, formal languages and computation.

The obtained results bring us to the following problems.

Problem 1. Study different modes of functioning for inductive Turing machines of the higher orders.

Problem 2. Study inductive Turing machines in which the control device is a more powerful automaton than a finite automaton.

Here we considered accepting and computing modes of inductive Turing machine brings us to the following problem.

Problem 3. For inductive Turing machines, study relations between the decision mode, accepting mode and computing mode of functioning.

It would be important to study properties of stabilizing computations in distributed systems. Grid automata provide the most advanced and general model of distributed systems (Burgin, 2005a).

Problem 4. Study properties of stabilizing computations for grid automata.

There are models of computation without explicit utilization of automata (cf., for example, (Milner, 1989; Lee & Sangiovanni-Vincentelli, 1996; Burgin & Smith, 2010)).

Problem 5. Study properties of stabilizing computations utilizing models of concurrent computational processes.

References

- Beros, A. A. (2013). Learning theory in the arithmetical hierarchy, preprint in mathematics. *math.LO/1302.7069* (electronic edition: <http://arXiv.org>).
- Büchi, J. Richard (1960). Weak second-order arithmetic and finite automata. *Z. Math. Logik und Grundl. Math.* **6**, 66–92.
- Burgin, M. (1999). Super-recursive algorithms as a tool for high performance computing. *Proceedings of the High Performance Computing Symposium, San Diego* **6**, 224–228.
- Burgin, M. (2003). Nonlinear phenomena in spaces of algorithms. *International Journal of Computer Mathematics* **80**(12), 1449–1476.
- Burgin, M. (2004). Algorithmic complexity of recursive and inductive algorithms. *Theoretical Computer Science* **317**(13), 31 – 60.
- Burgin, M. (2005a). *Super-recursive Algorithms*. Springer, New York.
- Burgin, M. (2005b). Superrecursive hierarchies of algorithmic problems. In: *Proceedings of the 2005 International Conference on Foundations of Computer Science, CSREA Press, Las Vegas*. pp. 31–37.
- Burgin, M. (2006). Algorithmic control in concurrent computations. *Proceedings of the 2006 International Conference on Foundations of Computer Science, CSREA Press, Las Vegas* **6**, 17–23.
- Burgin, M. (2007). Algorithmic complexity as a criterion of unsolvability. *Theoretical Computer Science* **383**(2/3), 244 – 259.
- Burgin, M. (2010a). Algorithmic complexity of computational problems. *International Journal of Computing & Information Technology*. **2**(1), 149–187.
- Burgin, M. (2010b). *Measuring power of algorithms, computer programs, and information automata*. Nova Science Publishers, New York.
- Burgin, M. and A. Klinger (2004). Experience, generations, and limits in machine learning. *Theoretical Computer Science* **317**(1/3), 71 – 91.
- Burgin, M. and B. Gupta (2012). Second-level algorithms, superrecursivity, and recovery problem in distributed systems. *Theory of Computing Systems* **50**(4), 694 – 705.
- Burgin, M. and E. Eberbach (2008). Cooperative combinatorial optimization: Evolutionary computation case study. *Biosystems* **91**(1), 34 – 50.
- Burgin, M. and E. Eberbach (2009a). On foundations of evolutionary computation: An evolutionary automata approach. *Handbook of Research on Artificial Immune Systems and Natural Computing: Applying Complex Adaptive Technologies*, Hongwei Mo, Ed., IGI Global, Hershey, Pennsylvania, pp. 342–360.
- Burgin, M. and E. Eberbach (2009b). Universality for Turing machines, inductive Turing machines and evolutionary algorithms. *Fundamenta Informaticae* **91**, 53–77.
- Burgin, M. and E. Eberbach (2010). Bounded and periodic evolutionary machines. *Proc. 2010 Congress on Evolutionary Computation (CEC'2010), Barcelona, Spain* pp. 1379–1386.
- Burgin, M. and E. Eberbach (2012). Evolutionary automata: Expressiveness and convergence of evolutionary computation. *The Computer Journal* **55**(9), 1023–1029.
- Burgin, M. and M. L. Smith (2010). A theoretical model for grid, cluster and internet computing. *Selected Topics in Communication Networks and Distributed Systems*, World Scientific, New York/London/Singapore pp. 485 – 535.
- Burgin, M. and N. Debnath (2004). Measuring software maintenance. *Proceedings of the ISCA 19th International Conference "Computers and their Applications", ISCA, Seattle, Washington* pp. 118–121.
- Burgin, M. and N. Debnath (2005). Complexity measures for software engineering. *J. Comp. Methods in Sci. and Eng.* **5**(1 Supplement), 127–143.
- Burgin, M. and N. Debnath (2009). Super-recursive algorithms in testing distributed systems. *Proceedings of the ISCA 24-th International Conference "Computers and their Applications", ISCA, New Orleans, Louisiana, USA* pp. 209–214.

- Burgin, M., N. Debnath and H. K. Lee (2009). Measuring testing as a distributed component of the software life cycle. *Journal of Computational Methods in Science and Engineering* **9**(Supplement 2/ 2009), 211–223.
- Burks, A. W. and J. B. Wright (1953). Theory of logical nets. *Proceedings of the IRE* **41**(10), 1357–1365.
- Calinescu, R., R. France and C. Ghezzi (2013). Editorial. *Computing* **95**(3), 165–166.
- Chadha, R., A. P. Sistla and M. Viswanathan (2009). Power of randomization in automata on infinite strings. In: *CONCUR 2009 - Concurrency Theory* (Mario Bravetti and Gianluigi Zavattaro, Eds.). Vol. 5710 of *Lecture Notes in Computer Science*. pp. 229–243. Springer, Berlin/Heidelberg.
- Gold, E. M. (1967). Language identification in the limit. *Information and Control* **10**(5), 447 – 474.
- Hopcroft, J. E., R. Motwani and J. D. Ullman (2001). *Introduction to Automata Theory, Languages, and Computation*. Addison Wesley, Boston/San Francisco/New York.
- Lee, E.A. and A. Sangiovanni-Vincentelli (1996). Comparing models of computation. In: *Computer-Aided Design, 1996. ICCAD-96. Digest of Technical Papers., 1996 IEEE/ACM International Conference on*. pp. 234–241.
- Milner, R. (1989). *Communication and Concurrency*. Prentice-Hall, Inc.. Upper Saddle River, NJ, USA.
- Rogers, Jr., Hartley (1987). *Theory of Recursive Functions and Effective Computability*. MIT Press. Cambridge, MA, USA.
- Sipser, M. (1996). *Introduction to the Theory of Computation*. 1st ed.. International Thomson Publishing.
- Thomas, Wolfgang (1990). Handbook of theoretical computer science (vol. b). Chap. Automata on Infinite Objects, pp. 133–191. MIT Press. Cambridge, MA, USA.



Second Hankel Determinant for Generalized Sakaguchi Type Functions

S. P. Vijayalakshmi^{a,*}, T. V. Sudharsan^b

^aDepartment of Mathematics, Ethiraj College, Chennai - 600 008, India

^bDepartment of Mathematics, SIVET College, Chennai - 600 073, India

Abstract

In this paper, we have obtained sharp upper bounds for the functional $|a_2a_4 - a_3^2|$ belonging to a new subclass of generalized Sakaguchi type functions introduced by (Frasin, 2010).

Keywords: Sakaguchi functions, subordination, Hankel determinant, starlike functions.
2010 MSC: 30C45, 30C50.

1. Introduction

Let A be the class of analytic functions of the form

$$f(z) = z + \sum_{n=2}^{\infty} a_n z^n; \quad (z \in \Delta := \{z \in \mathbb{C} : |z| < 1\}) \quad (1.1)$$

and S be the subclass of A consisting of univalent functions. For two functions $f, g \in A$, we say that the function $f(z)$ is subordinate to $g(z)$ in Δ and write $f < g$, or $f(z) < g(z)$; ($z \in \Delta$) if there exists an analytic function $w(z)$ with $w(0) = 0$ and $|w(z)| < 1$ ($z \in \Delta$), such that $f(z) = g(w(z))$, ($z \in \Delta$). In particular, if the function g is univalent in Δ , the above subordination is equivalent to $f(0) = g(0)$ and $f(\Delta) \subset g(\Delta)$.

Denote by S_{Σ}^* the subclass of S consisting of functions given by (1.1) satisfying $Re \left[\frac{zf'(z)}{f(z) - f(-z)} \right] > 0$ for $z \in \Delta$. These functions introduced by (Sakaguchi, 1959) are called functions starlike with respect to symmetric points.

*Corresponding author

Email addresses: vijishreekanth@gmail.com (S. P. Vijayalakshmi), tvsudharsan@rediffmail.com (T. V. Sudharsan)

Recently (Frasin, 2010) introduced and studied a generalized Sakaguchi type class $S(\alpha, s, t)$ if it satisfies

$$Re \left\{ \frac{(s-t)zf'(z)}{f(sz) - f(tz)} \right\} > \alpha \tag{1.2}$$

for some $0 \leq \alpha < 1$, $s, t \in C$ with $s \neq t$ and for all $z \in \Delta$. Also denote by $T(\alpha, s, t)$ the subclass of A consisting of all functions $f(z)$ such that $zf'(z) \in S(\alpha, s, t)$. The class $S(\alpha, 1, t)$ was introduced and studied by Owa et al. (Owa et al., 2005, 2007). If $t = -1$, the class $S(\alpha, 1, -1) \equiv S_s(\alpha)$ (Sakaguchi, 1959) is called Sakaguchi function of order α (see (Cho et al., 1993; Owa et al., 2005)), where as $S_s(0) = S_s^*$ (Sakaguchi, 1959).

Note that $S(\alpha, 1, 0) \equiv S^*(\alpha)$ and $T(\alpha, 1, 0) \equiv C(\alpha)$ which are, respectively, the familiar classes of starlike functions of order α ($0 \leq \alpha < 1$) and convex functions of order α ($0 \leq \alpha < 1$).

Mathur & Mathur (Trilok Mathur & Ruchi Mathur, 2012) investigated the classes $S_s^*(\phi, s, t)$ and $T(\phi, s, t)$ defined as follows.

Definition 1.1. Let $\phi(z) = 1 + B_1z + B_2z^2 + \dots$ be univalent starlike function with respect to 1 which maps the unit disk Δ onto a region in the right half plane which is symmetric with respect to the real axis, and let $B_1 > 0$. The function $f \in A$ is in the class $S_s^*(\phi, s, t)$ if

$$\left\{ \frac{(s-t)zf'(z)}{f(sz) - f(tz)} \right\} < \phi(z), \quad s \neq t.$$

Remark. $T(\phi, s, t)$ denotes the subclass of A consisting functions $f(z)$ such that $zf'(z) \in S_s^*(\phi, s, t)$. Observe that $S_s^*(\phi, 1, 0) \equiv S_s^*(\phi)$ and $T(\phi, 1, 0) \equiv C(\phi)$, which are the classes introduced and studied by Ma and Minda (Ma & Minda, 1994). Also note that $S_s^*(\phi, 1, -1) \equiv S_s^*(\phi)$, Shanmugam et al. (Shanmugham et al., 2006).

The q^{th} Hankel determinant for $q \geq 1$ and $n \geq 0$ is stated by Noonan and Thomas (Noonan & Thomas, 1976) as

$$H_q(n) = \begin{vmatrix} a_n & a_{n+1} & \dots & a_{n+q+1} \\ a_{n+1} & \dots & & \dots \\ \vdots & & & \vdots \\ a_{n+q-1} & \dots & & a_{n+2q-2} \end{vmatrix}$$

This determinant has also been considered by several authors. For example, Noor (Noor, 1983) determined the rate of growth of $H_q(n)$ as $n \rightarrow \infty$ for functions f given by (1.1) with bounded boundary. In particular, sharp upper bounds on $H_2(2)$ were obtained by the authors of articles (Hayami & Owa, 2010; Janteng et al., 2008; Kharudin et al., 2011; Noor, 1983; Selvaraj & Vasanthi, 2010) for different classes of functions.

Easily, one can observe that the Fekete-Szego functional is $H_2(1)$. Fekete-Szego then further generalised the estimate $|a_3 - \mu a_2^2|$ where μ is real and $f \in S$. For our discussion in this paper, we consider the Hankel determinant in the case $q = 2$ and $n = 2$,

$$H_2(2) = \begin{vmatrix} a_2 & a_3 \\ a_3 & a_4 \end{vmatrix}.$$

In the present investigation, we see the upper bound for the functional $|a_2a_4 - a_3^2|$ belonging to a new subclass $S_s^*(\phi, s, t)$ of generalized Sakaguchi type functions introduced by (Frasin, 2010).

2. Preliminary Results

Let P denote the class of functions

$$p(z) = 1 + c_1z + c_2z^2 + \dots \tag{2.1}$$

which are regular in Δ and satisfy $Re p(z) > 0, z \in \Delta$. Throughout this paper we assume that $p(z)$ is given by (2.1) and $f(z)$ is given by (1.1). To prove the main results we shall require the following lemmas.

Lemma 2.1. (Duren, 1983) Let $p \in P$, then $|c_k| \leq 2, k = 1, 2, \dots$ and the inequality is sharp.

Lemma 2.2. (Libera & Zlotkiewicz, 1982, 1983) Let $p \in P$, then

$$2c_2 = c_1^2 + x(4 - c_1^2) \tag{2.2}$$

and

$$4c_3 = c_1^3 + 2xc_1(4 - c_1^2) - x^2c_1(4 - c_1^2) + 2y(1 - |x|^2)(4 - c_1^2) \tag{2.3}$$

for some x, y such that $|x| \leq 1$ and $|y| \leq 1$.

3. Main Results

Theorem 3.1. If $f \in S_S^*(\phi, s, t)$, then

$$|a_2a_4 - a_3^2| \leq \frac{B_1}{(3 - s^2 - st - t^2)^2}, \text{ provided } s + t \neq 2. \tag{3.1}$$

The result obtained is sharp.

Proof. Let $f \in S_S^*(\phi, s, t)$. Then there exists a Schwarz function $w(z) \in A$ such that

$$\left\{ \frac{(s-t)zf'(z)}{f(sz) - f(tz)} \right\} = \phi(w(z)), \quad (z \in \Delta, s \neq t) \tag{3.2}$$

If $P_1(z)$ is analytic and has positive real part in Δ and $P_1(0) = 1$, then

$$P_1(z) = \frac{1 + w(z)}{1 - w(z)} = 1 + c_1z + c_2z^2 + \dots \tag{3.3}$$

From (3.3) we obtain

$$w(z) = \frac{c_1}{2}z + \frac{1}{2}\left(c_2 - \frac{c_1^2}{2}\right)z^2 + \dots \tag{3.4}$$

Let

$$p(z) = \frac{(s-t)zf'(z)}{f(sz) - f(tz)} = 1 + b_1z + b_2z^2 + \dots; \quad (z \in \Delta) \tag{3.5}$$

which gives

$$\begin{aligned} b_1 &= (2 - s - t)a_2, \\ b_2 &= (3 - s^2 - st - t^2)a_3 + (s + t)(s + t - 2)a_2^2, \\ b_3 &= (4 - s^3 - st^2 - s^2t - t^3)a_4 + 2(s^2 + st + t^2)(s + t - 1)a_2a_3 \\ &\quad + 2a_2^3(s + t)^2 - 3a_2a_3(s + t). \end{aligned}$$

Since $\phi(z)$ is univalent and $P < \phi$, therefore using (3.4) we obtain

$$\begin{aligned} P(z) = \phi(w(z)) &= 1 + \frac{B_1c_1}{2}z + \left\{ \frac{1}{2} \left(c_2 - \frac{c_1^2}{2} \right) B_1 + \frac{1}{4}c_1^2B_2 \right\} z^2 \\ &\quad + \left[\frac{B_1}{2} \left\{ 2c_3 + c_1 \left(\frac{c_1^2}{2} - c_2 \right) - c_1c_2 \right\} \right. \\ &\quad \left. + \frac{B_1c_1}{2} \left(c_2 - \frac{c_1^2}{2} \right) + \frac{B_3c_1^3}{8} \right] z^3 + \dots \end{aligned} \tag{3.6}$$

Now from (3.5) and (3.6) we have

$$\frac{(s - t)zf'(z)}{f(sz) - f(tz)} = 1 + \frac{B_1c_1}{2}z + \left\{ \frac{1}{2} \left(c_2 - \frac{c_1^2}{2} \right) B_1 + \frac{1}{4}c_1^2B_2 \right\} z^2 + \dots \tag{3.7}$$

On equating the coefficient of z , z^2 and z^3 in (3.7) we obtain

$$\begin{aligned} a_2 &= \frac{B_1c_1}{2(2 - s - t)}, \\ a_3 &= \frac{1}{(3 - s^2 - st - t^2)} \left\{ \frac{1}{2} \left(c_2 - \frac{c_1^2}{2} \right) B_1 + \frac{1}{4}c_1^2B_2 + \frac{(s + t)B_1^2c_1^2}{4(2 - s - t)} \right\}, \\ a_4 &= \frac{1}{(4 - s^3 - st^2 - s^2t - t^3)} \left[B_1c_3 + c_1^3 \left[\frac{B_1}{4} - \frac{B_2}{4} + \frac{B_3}{8} - \frac{B_1^3(s + t)^2}{4(2 - s - t)^3} \right. \right. \\ &\quad \left. \left. + \frac{(s + t + 2(s^2 + st + t^2))}{16(2 - s - t)(3 - s^2 - st - t^2)} \left\{ B_1B_2 - B_1^2 + \frac{B_1^3}{(2 - s - t)} \right\} \right] \right. \\ &\quad \left. + c_1c_2 \left[-\frac{3B_1}{2} + \frac{B_2}{2} \right. \right. \\ &\quad \left. \left. + \frac{B_1^2}{8(2 - s - t)(3 - s^2 - st - t^2)} [3(s + t) - 2(s^2 + st + t^2)(s + t - 1)] \right] \right]. \end{aligned}$$

Thus we have,

$$\begin{aligned}
 |a_2a_4 - a_3^2| &= \left| \frac{B_1c_1}{2P_1} \left\{ B_1c_3 + c_1^3 \left[\frac{2(B_1 - B_2) + B_3}{8} - \frac{B_1^3(s+t)^2}{4(2-s-t)^3} \right. \right. \right. \\
 &\quad \left. \left. + \frac{3(s+t) - 2(s^2 + st + t^2)(s+t-1)}{16(2-s-t)(3-s^2-st-t^2)} \left\{ B_1B_2 - B_1^2 + \frac{B_1^3}{(2-s-t)} \right\} \right] \right\} \\
 &\quad + c_1c_2 \left[-\frac{3B_1}{2} + \frac{B_2}{2} + \frac{B_1^2}{8(2-s-t)(3-s^2-st-t^2)} \right. \\
 &\quad \left. [3(s+t) - 2(s^2 + st + t^2)(s+t-1)] \right] \\
 &\quad \left. - \frac{1}{P_3} \left\{ c_1^4f_1(s,t) + c_2^2g_1(s,t) + \frac{c_1^2B_1^2}{16} + \frac{c_2^2B_1^2}{4} \right\} \right|.
 \end{aligned}$$

Suppose now that $c_1 = c$. Since $|c| = |c_1| \leq 2$, using the Lemma 2.1, we may assume without restriction $c \in [0, 2]$. Substituting for c_2 and c_3 , from Lemma 2.2 and applying the triangle inequality with $\rho = |x|$, we obtain

$$\begin{aligned}
 |a_2a_4 - a_3^2| &\leq c^4 \left[\frac{f(s,t)}{P_1} + \frac{B_1g(s,t)}{4P_1} - \frac{f_1(s,t)}{P_3} - \frac{g_1(s,t)}{2P_3} - \frac{B_1}{16P_3} \right] \\
 &\quad + c^2(4 - c^2)\rho \left[\frac{B_1}{4P_1} + \frac{B_1^2}{4P_1} - \frac{B_1^2\rho}{8P_1} - \frac{g_1(s,t)}{2P_3} - \frac{B_1}{8P_3} \right] \\
 &\quad + \frac{c^2B_1^2}{16P_3} + \frac{B_1^2c(4 - c^2)(1 - \rho^2)}{4P_1} + \frac{B_1\rho^2(4 - c^2)^2}{16P_3} \\
 &= F(\rho)
 \end{aligned} \tag{3.8}$$

where,

$$\begin{aligned}
 P_1 &= (2 - s - t)(4 - s^3 - st^2 - s^2t - t^3), \\
 P_2 &= 8(2 - s - t)(3 - s^2 - st - t^2), \\
 P_3 &= (3 - s^2 - st - t^2)^2,
 \end{aligned}$$

$$\begin{aligned}
 f_1(s,t) &= \frac{B_2^2}{16} + \frac{(s+t)^2B_1^4}{16(2-s-t)^2} - \frac{B_1B_2}{8} + \frac{B_1^2B_2(s+t)}{8(2-s-t)} + \frac{(s+t)B_1^3}{8(2-s-t)}, \\
 g_1(s,t) &= \frac{(s+t)B_1^3}{4(2-s-t)} + \frac{B_1B_2}{4} - \frac{B_1^2}{4}, \\
 f(s,t) &= \frac{B_1^2}{4} - \frac{B_1B_2}{8} + \frac{B_1B_3}{8} - \frac{B_1^4(s+t)^2}{8(2-s-t)^3} \\
 &\quad + \frac{[(B_1B_2 - B_1^2)(2-s-t)] + B_1^3[3(s+t) - 2(s^2 + st + t^2)(s+t-1)]}{32(2-s-t)^2(3-s^2-st-t^2)}, \\
 g(s,t) &= -B_1 + \frac{B_2}{2} - \frac{B_1^2(3(s+t) - 2(s^2 + st + t^2)(s+t-1))}{2},
 \end{aligned}$$

with $\rho = |x| \leq 1$. Furthermore

$$F'(\rho) = c^2(4 - c^2) \left[\frac{B_1}{4P_1} + \frac{B_1^2}{4P_1} - \frac{B_1^2\rho}{8P_1} - \frac{g_1(s, t)}{2P_3} - \frac{B_1}{8P_3} \right] + \frac{B_1^2c\rho(4 - c^2)(4 - c)}{8P_1} + \frac{B_1^2(4 - c^2)^2\rho}{8P_3}.$$

For a $c \in [0, 2]$, $F(\rho) \leq F(1)$, that is

$$\begin{aligned} |a_2a_4 - a_3^2| &\leq c^4 \left[\frac{f(s, t)}{P_1} + \frac{B_1g(s, t)}{4P_1} - \frac{f_1(s, t)}{P_3} - \frac{g_1(s, t)}{2P_3} - \frac{B_1}{16P_3} \right] \\ &\quad + c^2(4 - c^2) \left[\frac{B_1}{4P_1} + \frac{B_1^2}{4P_1} - \frac{B_1^2}{8P_1} - \frac{g_1(s, t)}{2P_3} - \frac{B_1}{8P_3} \right] \\ &\quad + \frac{c^2B_1^2}{16P_3} + \frac{B_1(4 - c^2)^2}{16P_3} \\ &= G(c). \end{aligned}$$

By elementary calculus we have $G''(c) \leq 0$ for $0 \leq c \leq 2$ and $G(c)$ has real critical point at $c = 0$. Thus the upper bound of $F(\rho)$ corresponds to $\rho = 1$ and $c = 0$. Thus the maximum of $G(c)$ occurs at $c = 0$. Hence,

$$|a_2a_4 - a_3^2| \leq \frac{B_1}{(3 - s^2 - st - t^2)^2}$$

If $p(z) \in P$ with $c_1 = c = 0$, $c_2 = 2$, $c_3 = 1$, then we obtain

$p(z) = (1 - z) + \frac{z}{(1 - z)^2} = 1 + 2z^2 + z^3 + \dots \in P$. The result is sharp for the functions defined by

$$\left\{ \frac{(s - t)zf'(z)}{f(sz) - f(tz)} \right\} = \phi(z), \quad s \neq t$$

and

$$\left\{ \frac{(s - t)zf'(z)}{f(sz) - f(tz)} \right\} = \phi(z^2), \quad s \neq t.$$

□

Remark. If $f \in S_s^*(\phi, 1, -1)$, then

$$|a_2a_4 - a_3^2| \leq \frac{B_1}{2}.$$

Since $f(z) \in T(\phi, s, t)$ if and only if $zf'(z) \in S_s^*(\phi, s, t)$, proceeding on similar lines as in Theorem 3.1 we obtain the upper bound for the functional $|a_2a_4 - a_3^2|$ belonging to the class $T(\phi, s, t)$ which is stated below without proof.

Theorem 3.2. *If $f \in T(\phi, s, t)$, then*

$$|a_2a_4 - a_3^2| \leq \frac{B_1^2}{9(3 - s^2 - st - t^2)^2}, \quad \text{provided } s + t \neq 2. \tag{3.9}$$

The result obtained is sharp.

Remark. If $f \in T(\phi, 1, -1)$, then

$$|a_2a_4 - a_3^2| \leq \frac{B_1^2}{36}.$$

Acknowledgement

The authors thank the referee for the valuable comments and suggestions to improve the presentation of the paper.

References

- Cho, N. E., O. S. Kwon and S. Owa (1993). Certain subclasses of Sakaguchi functions. *SEA Bull. Math.* **17**, 121–126.
- Duren, P. L. (1983). *Univalent functions*. Springer Verlag, New York Inc.
- Frasin, B. A. (2010). Coefficient inequalities for certain classes of Sakaguchi type functions. *Int. J. Nonlinear Sci.* **10**(2), 206–211.
- Hayami, T. and S. Owa (2010). Generalized Hankel determinant for certain classes. *Int. Journal of Math. Analysis* **4**(52), 2473–2585.
- Janteng, A., S. A. Halim and M. Darus (2008). Estimate on the second Hankel functional for functions whose derivative has a positive real part. *Journal of Quality Measurement and Analysis* **4**(1), 189–195.
- Kharudin, N., A. Akbarally, D. Mohamad and S. C. Soh (2011). The second Hankel determinant for the class of close to convex functions. *European Journal of Scientific Research* **66**(3), 421–427.
- Libera, R. J. and E. J. Zlotkiewicz (1982). Early coefficients of the inverse of a regular convex function. *Proc. Amer. Math. Soc.* **85**(2), 225–230.
- Libera, R. J. and E. J. Zlotkiewicz (1983). Coefficient bounds for the inverse of a function with derivative in p . *Proc. Amer. Math. Soc.* **87**(2), 251–257.
- Ma, W. and D. Minda (1994). A unified treatment of some special classes of univalent functions. In: *Proceedings of Conference of Complex Analysis* (Z. Li, F. Ren, L. Yang and S. Zhang, Eds.). Intenational Press. pp. 157–169.
- Noonan, J. W. and D. K. Thomas (1976). On the second Hankel determinant of areally mean p -valent functions. *Transactions of the Americal Mathematical Society* **223**(2), 337–346.
- Noor, K. I. (1983). Hankel determinant problem for the class of functions with bounded boundary rotation. *Rev. Roum. Math. Pures Et Appl.* **28**(c), 731–739.
- Owa, S., T. Sekine and R. Yamakawa (2005). Notes on Sakaguchi type functions. *RIMS Kokyuroku* **1414**, 76–82.
- Owa, S., T. Sekine and R. Yamakawa (2007). On Sakaguchi type functions. *Appl Math. Comput.* **187**, 356–361.
- Sakaguchi, K. (1959). On a certain univalent mapping. *J. Math. Soc. Japan* **11**, 72–75.
- Selvaraj, C. and N. Vasanthi (2010). Coefficient bounds for certain subclasses of close-to-convex functions. *Int. Journal of Math. Analysis* **4**(37), 1807–1814.
- Shanmugham, T. N., C. Ramachandran and V. Ravichandran (2006). Fekete-Szegö problem for a subclasses of starlike functions with respect to symmetric points. *Bull. Korean Math. Soc.* **43**(3), 589–598.
- Trilok Mathur and Ruchi Mathur (2012). Fekete-Szegö inequalities for generalized Sakaguchi type functions. In: *Proceedings of the World Congress on Engineering, WCE 2012*. Vol. 1. London, U.K.



L_p - Approximation of Analytic Functions on Compact Sets Bounded by Jordan Curves

Devendra Kumar^{a,*}, Vandna Jain^b

^a*Department of Mathematics, M.M.H. College, Ghaziabad-201 001, U.P. India*

^b*Department of Mathematics, Punjab Technical University, Jalandhar (Pb.), India*

Abstract

This paper is concerned with functions analytic on compact sets bounded by Jordan curves having rapidly increasing maximum modulus such that order of function is infinite. To study the precise rates of growth of such functions the concept of index has been used. The q -order and lower q -order of analytic functions have been obtained in terms of L_p -approximation error. Our results improve and refine the results of Andre Giroux (Giroux, 1980) and Kapoor and Nautiyal (Kapoor & Nautiyal, 1982) for non entire case.

Keywords: L_p -approximation error, index- q , transfinite diameter, Faber series.

2010 MSC: Primary 30D10; Secondary 41A10.

1. Introduction

Let D be a compact set containing at least two points such that its complement D' with respect to the extended complex plane is a simply connected domain containing the point at infinity. In view of Riemann mapping theorem, there exists a one-one analytic function $z = \varphi(w)$ which maps $\{w : |w| > 1\}$ conformly onto D' such that $\varphi(\infty) = \infty$ and $\varphi'(\infty) > 0$. Thus, in a neighborhood of infinity, the function has the expansion

$$z = \varphi(w) = d \left[w + d_0 + \frac{d_{-1}}{w} + \dots \right]$$

where the number $d > 0$ is called the transfinite diameter of D . If we define $\eta(w) = \varphi(w/d)$, then η maps $\{w : |w| > d\}$ onto D' in a one-one conformal manner. If $w = \Omega(z)$ is the inverse function of η then $\Omega(\infty) = \infty$ and $\lim_{z \rightarrow \infty} \Omega(z)/z = 1$.

*Corresponding author

Email addresses: d_kumar001@rediffmail.com (Devendra Kumar), vandnajain.mittal@gmail.com (Vandna Jain)

For $1 \leq p < \infty$, let $L_p(D)$ denote the space of analytic functions f in D such that

$$\|f\|_{D,p} = \left(\frac{1}{A} \int_D |f(z)|^p dx dy \right)^{1/p} < \infty, \text{ where } A \text{ is the area of } D.$$

Let L_r is an analytic Jordan curve for each $r > d$. If D_r denotes the domain bounded by L_r , then $D \subset D_r$ for each $r > d$. Let $H(\overline{D}; R)$ denotes the class of all functions that are regular in D_R with a singularity on L_R ($d < R < \infty$). Since $\overline{D} \subset D_R$ for $R > d$ it follows that every $f \in H(\overline{D}; R)$ is analytic in \overline{D} and so $\int_D |f(z)|^p dx dy < \infty$ and $f \in L_p(D)$.

We now prove the following:

Theorem 1.1. *Every $f \in H(\overline{D}; R)$ can be represented by the Faber series*

$$f(z) = \sum_{n=0}^{\infty} a_n P_n(z), \quad z \in D_R \quad (1.1)$$

with

$$a_n = \frac{1}{2\pi i} \int_{|\xi|=1} f(\eta(\xi)) \xi^{-n-1} d\xi, \quad d < r < R \quad (1.2)$$

if and only if

$$\limsup_{n \rightarrow \infty} |a_n|^{1/n} = \frac{1}{R}. \quad (1.3)$$

The series in (1.1) converges absolutely and uniformly on every compact subset of D_R and diverge outside L_R .

Proof. Let $f \in H(\overline{D}; R)$. If $z \in D_R$, then $z \in D_r$ for some r satisfying $d < r < R$. Using Cauchy's integral formula,

$$f(z) = \frac{1}{2\pi i} \int_{L_r} \frac{f(t) dt}{t-z} = \frac{1}{2\pi i} \int_{|\xi|=r} \frac{f(\eta(\xi)) \eta'(\xi)}{n(\xi) - z} d\xi = \frac{1}{2\pi i} \int_{|\xi|=r} \left(\sum_{n=0}^{\infty} \frac{f(\eta(\xi))}{\xi^{n+1}} P_n(z) \right) d\xi.$$

Since the series under the integral sign converges uniformly on $|\xi| = r$, it can be integrated term by term. Thus we have

$$f(z) = \sum_{n=0}^{\infty} a_n P_n(z), \quad z \in D_R.$$

If

$$\overline{M}(r, f) = \max_{|\xi|=r} |f(\eta(\xi))|,$$

then (1.2) gives

$$|a_n| \leq \frac{\overline{M}(r, f)}{r^n}, \quad n = 0, 1, 2, \dots, \quad (1.4)$$

which are analogous of Cauchy's inequality for Taylor series. From (1.4), we have

$$\limsup_{n \rightarrow \infty} |a_n|^{1/n} \leq \frac{1}{r}.$$

Since this holds for every r satisfying $d < r < R$, we have

$$\limsup_{n \rightarrow \infty} |a_n|^{1/n} \leq \frac{1}{R}.$$

We now show that inequality does not holds in the above relation. If

$$\limsup_{n \rightarrow \infty} |a_n|^{1/n} = \frac{1}{R_0} < \frac{1}{R},$$

let r satisfy $d < r < R_0$; then for every ε such that $0 < \varepsilon < R_0 - r$, we have

$$|a_n| < \frac{1}{(R_0 - \varepsilon/2)^n} < \frac{1}{(r + \varepsilon/2)^n} \text{ for } n \geq n_0.$$

On the other hand, for every $z \in L_r$, we have $|P_n(z)| < (r + \varepsilon/4)^n$ for $n \geq n_1$.

Thus, for all $z \in L_r$, $|a_n P_n(z)| < \left(\frac{r+\varepsilon/4}{r+\varepsilon/2}\right)^n < 1$ for $n \geq \max(n_0, n_1)$.

The above inequality shows that the series $\sum_{n=0}^{\infty} a_n P_n(z)$ converges uniformly on L_r and hence on D_r . Since this is true for every r satisfying $d < r < R_0$, it follows that the series $\sum_{n=0}^{\infty} a_n P_n(z)$ converges uniformly on every compact subset of D_{R_0} to a function, $F(z)$, say. The function $F(z)$ must be regular in D_{R_0} since each term of the series is a regular function. However, on $D_R \subset D_{R_0}$, the series converges to $f(z)$ so that $F(z)$ is the analytic continuation of $f(z)$ to D_{R_0} . Since this contradicts the hypothesis that f has a singular point on $L_R (R < R_0)$, we must have (1.3).

To show that the series (1.3) diverges outside L_R , let z_0 lie outside L_R . Then we have $|\Omega(z_0)| > R$. In view of $\lim_{n \rightarrow \infty} |P_n(z)|^{1/n} = |\Omega(z)|$ and (1.3) we obtain $\lim_{n \rightarrow \infty} \sup |a_n P_n(z_0)|^{1/n} > 1$, showing that the series $\sum_{n=0}^{\infty} a_n P_n(z_0)$ diverges.

Conversely, if (1.3) holds, then as above, we can show that the series $\sum_{n=0}^{\infty} a_n P_n(z)$ converges uniformly on compact subsets of D_R to a regular function, $f(z)$, say. If $f(z)$ had no singular point on L_R then it would be possible to extend $f(z)$ analytically to a bigger domain D_{R_0} , say. But then the first part of the theorem would give that

$$\lim_{n \rightarrow \infty} \sup |a_n|^{1/n} \leq 1/R_0 < 1/R, \text{ a contradiction. Hence the proof is completed.} \quad \square$$

In view of above theorem, there exists a sequence of polynomials converging uniformly on compact subsets of D_R to $f(z)$, it follows that this sequence converges in the norm of $L_p(D)$ also to f . If p_{n-1} denotes the collection of all polynomials of degree not exceeding $n - 1$ and we set

$$E_n^p(f) = \inf_{g \in p_{n-1}} \|f - g\|_{D,p}, \quad n = 1, 2, \dots,$$

then it is clear that $E_n^p(f)$ is a non increasing sequence tending to zero as $n \rightarrow \infty$. Our next theorem holds for $1 \leq p < \infty$.

Theorem 1.2. *If $f \in H(\overline{D}; R)$, then*

$$\limsup_{n \rightarrow \infty} [E_n^p(f)]^{1/n} = \frac{d}{R}. \tag{1.5}$$

Proof. If $f \in H(\bar{D}; R)$ we have, by (1.5)

$$E_n^p(f) = \inf_{g \in p_{n-1}} \left(\int \int_D |f(z) - g(z)|^p dx dy \right)^{1/p} \leq \left(\int \int_D |f(z) - Q_{n-1}(z)|^p dx dy \right)^{1/p} \leq A^{1/p} \max_{z \in \bar{D}} |f(z) - Q_{n-1}(z)|,$$

where $Q_{n-1}(z)$ is the polynomial of degree not exceeding $n - 1$ and A is the area of domain D . Using a result of (Markushevich, 1967), p. 114 for $d < r' < r < R$ and $n > n_0$ we get

$$E_n^p(f) \leq A^{1/p} \bar{M}(r, f) \left(\frac{r'}{r - r'} \right) (r'/r)^n \tag{1.6}$$

where $\bar{M}(r, f) = \max_{z \in L_r} |f(z)|$. This leads to $\lim_{n \rightarrow \infty} \sup [E_n^p(f)]^{1/n} \leq r'/r$.

Since the above relation holds for all r', r satisfying $d < r' < r < R$, we must have

$$\limsup_{n \rightarrow \infty} [E_n^p(f)]^{1/n} \leq \frac{d}{R}. \tag{1.7}$$

To obtain the reverse inequality in (1.7), we note that, since every $f \in H(\bar{D}; R)$ is in $H_2(D)$, there exists a closed orthonormal system $\{\chi_n(z)\}_{n=0}^\infty$ of polynomials in $H_2(D)$ such that f can be represented by its Fourier series with respect to the system $\{\chi_n(z)\}_{n=0}^\infty$ that converges uniformly on compact subsets of D_R to f . Thus

$$f(z) = \sum_{n=0}^\infty a_n \chi_n(z) \quad z \in D_R, \tag{1.8}$$

where $a_n = \int \int_D f(z) \bar{\chi}_n(z) dx dy$.

If $g \in p_{n-1}$, then

$$|a_n| = \left| \int \int_D (f(z) - g(z)) \bar{\chi}_n(z) dx dy \right| \leq \left(\int \int_D |f(z) - g(z)|^p dx dy \right)^{1/p} \left(\int \int_D |\bar{\chi}_n(z)|^{p/p-1} dx dy \right)^{1-1/p}.$$

Using (1.3) and the fact that the above inequality holds for every $g \in p_{n-1}$, we get, for $r^* > d$, $|a_n| \leq E_n^p(f) \cdot \bar{M}(r, f) \left(\frac{r^*}{d} \right)^n A^{1-1/p}$, by (1.7) we get $\lim_{n \rightarrow \infty} \sup [E_n^p(f)]^{1/n} \geq \frac{d}{r^*} \lim_{n \rightarrow \infty} \sup |a_n|^{1/n} = \frac{d^2}{r^* R}$.

Since the above inequality is valid for every $r^* > d$, we must have

$$\limsup_{n \rightarrow \infty} [E_n^p(f)]^{1/n} \geq \frac{d}{R}. \tag{1.9}$$

Combining (1.7) and (1.9) we get (1.5). □

2. Fast Growth and Approximation Errors

We now obtain relations that indicate how the growth of an $f \in H(\overline{D}; R)$ depends on $E_n^p(f)$ and vice versa.

For function $f \in H(\overline{D}; R)$, set

$$\rho_R(q) = \limsup_{r \rightarrow R} \frac{\log^{[q]} \overline{M}(r, f)}{\log(Rr/(R-r))} \tag{2.1}$$

where $\log^{[0]} \overline{M}(r, f) = \overline{M}(r, f)$ and $\log^{[q]} \overline{M}(r, f) = \log(\log^{[q-1]} \overline{M}(r, f))$, $q = 1, 2, \dots$. To avoid the trivial cases we shall assume throughout that $\overline{M}(r, f) \rightarrow \infty$ as $r \rightarrow R$.

Definition 2.1. A function $f \in H(\overline{D}; R)$, is said to have the index q if $\rho_R(q) < \infty$ and $\rho_R(q-1) = \infty$, $q = 1, 2, \dots$. If q is the index of $f(z)$, then $\rho_R(q)$ is called the q -order of f .

Definition 2.2. A function $f \in H(\overline{D}; R)$ and having the index $-q$ is called to have lower q -order $\lambda_R(q)$ if

$$\lambda_R(q) = \liminf_{r \rightarrow R} \frac{\log^{[q]} \overline{M}(r, f)}{\log(Rr/(R-r))}, q = 1, 2, \dots \tag{2.2}$$

Definition 2.3. A function $f \in H(\overline{D}; R)$ and having the index q is said to be of regular q -growth if $\rho_R(q) = \lambda_R(q)$, $q = 1, 2, \dots$, $f(z)$ is said to be of irregular q -growth if $\rho_R(q) > \lambda_R(q)$, $q = 1, 2, \dots$.

In 1980 Andre Giroux (Giroux, 1980) obtained necessary and sufficient conditions, in terms of the rate of decrease of the approximation error $E_n^p(f)$, such that $f \in L_p(D)$, $2 \leq p \leq \infty$, has an analytic continuation as an entire function having finite growth parameters. In 1982 Kapoor and Nautiyal (Kapoor & Nautiyal, 1982) considered the approximation error $E_n^p(f)$ on a Caratheodory domain and had extended the results of Giroux for the case $1 \leq p < 2$. All these results do not give any specific information about the growth when function is not entire and for the functions having rapidly increasing maximum modulus such that order of function is infinite. Although, Kumar (Kumar, 2004, 2007b,a, 2010, 2011, 2013) and Kumar and Mathur (Kumar & Amit, 2006) obtained some results in this direction but our results are different from all those of above papers.

In this paper an attempt has been made to study the growth of $f \in H(\overline{D}; R)$ involving $E_n^p(f)$ for $1 \leq p < \infty$ when f is not entire and having $\overline{M}(r, f) \rightarrow \infty$ as $r \rightarrow R$. To obtain the results in general setting we shall assume that $f \in H(\overline{D}; R)$ is represented by the gap power series $f(z) = \sum_{n=0}^{\infty} a_n z^{\lambda_n}$ where $\{\lambda_n\}_{n=0}^{\infty}$ is strictly increasing sequence of integers and $a_n \neq 0$ for all n .

Theorem 2.1. If $f \in H(\overline{D}; R)$ having the index $-q$ and q -order $\rho_R(q)$, then

$$\rho_R(q) + A(q) = \limsup_{n \rightarrow \infty} \frac{\log^{[q-1]} \lambda_n}{\log \lambda_n - \log^+ \log^+ ((E_{\lambda_n}^p(f))(R/d)^{\lambda_n})}, q = 2, 3, \dots, \tag{2.3}$$

where, $A(q) = 1$ if $q = 2$ and $A(q) = 0$ if $q = 3, 4, \dots$

Proof. If $f \in H(\bar{D}; R)$ is of q -order $\rho_R(q)$, given $\varepsilon > 0$, there exists $r_0(\varepsilon)$ such that, for $r_0 < r < R$, we have

$$\log^{[q-1]} \bar{M}(r, f) < \left(\frac{Rr}{R-r}\right)^{\rho_R(q)+\varepsilon}.$$

Using inequality (1.6) for $n > n_0$ and $d < r' < r < R, r_0 < r'$, we obtain

$$\begin{aligned} \log E_{\lambda_n}^{(p)}(f)(R/d)^{\lambda_n} &< \frac{1}{p} \log A + \exp^{[q-2]} \left(\frac{Rr}{R-r}\right)^{\rho_R(q)+\varepsilon} + \log \frac{r'}{r-r'} + \lambda_n \log \frac{r'}{d} + \lambda_n \log \frac{R}{r} \\ &< \frac{1}{p} \log A + \exp^{[q-2]} \left(\frac{Rr}{R-r}\right)^{\rho_R(q)+\varepsilon} + \log \frac{r'}{r-r'} + \lambda_n \left(\frac{r'-d}{d}\right) + \lambda_n \left(\frac{R-r}{r}\right). \end{aligned} \tag{2.4}$$

Let us consider r such that

$$\begin{aligned} \frac{Rr}{R-r} &= \left(\log^{[q-2]}(\lambda_n R/\rho_R(q) + \varepsilon)\right)^{1/\rho_R(q)+A(q)+\varepsilon}, \text{ and} \\ r' &= \lambda d + (1-\lambda)(Rd/r), 0 < \lambda < 1. \end{aligned} \tag{2.5}$$

For $q = 2$ the inequality (2.4) with (2.5) gives for $n > n_1$,

$$\log^+ E_{\lambda_n}^p(f)(R/d)^{\lambda_n} < M(\lambda_n R)^{(\rho_R(2)+\varepsilon)/(\rho_R(2)+1+\varepsilon)}$$

where M is a constant. It gives

$$\limsup_{n \rightarrow \infty} \frac{\log^+ \log^+ E_{\lambda_n}^p(f)(R/d)^{\lambda_n}}{\log \lambda_n} \leq \frac{\rho_R(2)}{\rho_R(2) + 1}. \tag{2.6}$$

For $q = 3, 4, \dots$, the inequality (2.4) gives for $n > n_1$, with (2.5) that

$$\log^+ E_{\lambda_n}^p(f)(R/d)^{\lambda_n} < \exp^{[q-2]} \left(\log^{[q-2]}(\lambda_n R/\rho_R(q) + \varepsilon)\right) [1 + 0(1)],$$

from which a simple calculation would yield

$$\rho_R(q) \geq \limsup_{n \rightarrow \infty} \frac{\log^{[q-1]} \lambda_n}{\log \lambda_n - \log^+ \log^+ E_{\lambda_n}^p(f)(R/d)^{\lambda_n}}. \tag{2.7}$$

To prove that reverse inequality, use (1.8) and (1) to get, for $z \in \bar{D}_r, d < r^* < r < R$,

$$\begin{aligned} |f(z)| &\leq \sum_{n=0}^{\infty} |a_n| |\chi_{\lambda_n}(z)| \\ &\leq MA^{1-1/p} \sum_{n=0}^{\infty} E_n^p(f) \left(\frac{r^*}{d}\right)^n |\chi_{\lambda_n}(z)|. \end{aligned}$$

It is known that for any $r^* > d$ there exists a constant M^l such that

$$|\chi_n(z)| \leq M'(r^*/d)^n, n = 0, 1, 2, \dots, z \in \bar{D}.$$

Now for $d < r^* < r < R$, we get

$$\overline{M}(r, f) \leq M^* M^l A^{1-1/p} \sum_{n=0}^{\infty} E_n^p(f)(R/d)^{\lambda_n} \left(\frac{r^{*2} r}{d^2 R}\right)^{\lambda_n}.$$

Taking $r^* = \sqrt[\lambda]{\lambda + (1 - \lambda)(R/r)}$, $0 < \lambda < 1$, the above inequality gives

$$\overline{M}(r, f) \leq BM \left(\frac{\lambda r + (1 - \lambda)R}{R}, G \right), \tag{2.8}$$

where B is constant, $G(s) = \sum_{n=0}^{\infty} E_n^p(f)(R/d)^{\lambda_n} s^{\lambda_n}$ and $M(t, G) = \max_{|s|=t} |G(s)|$. It can be easily seen that $G(s)$ is analytic in $|s| < 1$. If the q -order of $G(s)$ in unit disc is $\rho_0(q)$ then

$$\rho_R(q) = \limsup_{r \rightarrow R} \frac{\log^{[q]} \overline{M}(r, f)}{\log(Rr/(R - r))} \leq \limsup_{r \rightarrow R} \frac{\log^{[q]} M((\lambda r + (1 - \lambda)R)/R), G}{\log(Rr/(R - r))} = \rho_0(q).$$

Applying Theorem 1 of (Kapoor & Gopal, 1979) for $\rho_0(q)$, we obtain

$$\rho_R(q) + A(q) \leq \limsup_{n \rightarrow \infty} \frac{\log^{[q-1]} \lambda_n}{\log \lambda_n - \log^+ \log^+ E_{\lambda_n}^p(f)(R/d)^{\lambda_n}} q = 2, 3, \dots, \tag{2.9}$$

combining (2.6), (2.7) and (2.9) we get (2.3) i.e., the proof of theorem is completed. □

Theorem 2.2. Let $f \in H(\overline{D}; R)$ having the index- q , q -order $\rho_R(q)$ and lower q -order $\beta_R(q)$ ($0 \leq \beta_R(q) \leq \infty$), then for any increasing sequence $\{n_k\}$ of natural numbers,

$$\beta_R(q) + A(q) \geq \liminf_{k \rightarrow \infty} \frac{\log^{[q-1]} \lambda_{n_{k-1}}}{\log \lambda_{n_k} - \log^+ \log^+ E_{\lambda_{n_k}}^p(f)(R/d)^{\lambda_{n_k}}}. \tag{2.10}$$

Proof. Let the right hand side of (2.10) be δ . Without loss of generality we can assume $\delta > 0$. For any ε such that $0 < \varepsilon < \delta$, and for all $k > k_0 = k_0(\varepsilon)$, we get

$$\log^+ E_{\lambda_{n_k}}^p(f)(R/d)^{\lambda_{n_k}} > \lambda_{n_k} (\log^{[q-2]} \lambda_{n_{k-1}})^{-1/(\delta-\varepsilon)}.$$

Choosing a sequence r_k such that $r_k \leq r \leq r_{k+1}$, where $\frac{R-r_k}{r_k} = \frac{1}{e} (\log^{[q-2]} \lambda_{n_{k-1}})^{1/(\delta-\varepsilon)}$.

Using inequality (1.6) we obtain

$$\begin{aligned} \log \overline{M}(r, f) &\geq \log E_{\lambda_{n_k}}^p(f)(R/d)^{\lambda_{n_k}} - \frac{1}{p} \log A - \log \frac{r'}{r - r'} - \lambda_{n_k} \log \frac{r'}{d} - \lambda_{n_k} \log R/r \\ &\geq \log E_{\lambda_{n_k}}^p(f)(R/d)^{\lambda_{n_k}} - \frac{1}{p} \log A - \log \frac{r'}{r_k - r'} - \lambda_{n_k} \log \frac{r'}{d} - \lambda_{n_k} \log R/r_k \\ &> \lambda_{n_k} (\log^{[q-2]} \lambda_{n_{k-1}})^{-1/(\delta-\varepsilon)} - \lambda_{n_k} \left(\frac{R - r_k}{r_k}\right) = (1 - 1/e) \lambda_{n_k} (\log^{[q-2]} \lambda_{n_{k-1}})^{-1/(\delta-\varepsilon)} \\ &> (e - 1) \left(\frac{R - r}{r}\right) \exp^{[q-2]} \left(e \left(\frac{R - r}{r}\right)\right)^{-1/(\delta-\varepsilon)}. \end{aligned}$$

Now after a simple calculation the above estimate yields

$$\beta_R(q) + A(q) \geq \delta. \tag{2.11}$$

Hence the proof is completed. □

To prove our next theorem we need the following lemma.

Lemma 2.1. Let $f(z) = \sum_{n=0}^{\infty} a_n z^{\lambda_n}$ be analytic in unit disc having index q , q -order $\rho(q) > 0$ and lower q -order $\beta(q)$. Further, let $\varphi(n) \equiv |a_n/a_{n+1}|^{1/(\lambda_{n+1}-\lambda_n)}$ forms a non-decreasing function of n for $n > n_0$ and

$$\beta(q) + A(q) = \liminf_{r \rightarrow 1} \frac{\log^{[q-1]} \nu(r)}{-\log(1-r)}. \tag{2.12}$$

Then

$$\beta(q) + A(q) = \liminf_{n \rightarrow \infty} \frac{\log^{[q-1]} \lambda_n}{\log \lambda_n - \log^+ \log^+ |a_n|}$$

where for $|z| = r, \mu(r) = \max_{n>0} \{|a_n| r^{\lambda_n}\}, \nu(r) = \max\{\lambda_n : \mu(r) = |a_n| r^{\lambda_n}\}, 0 < r < 1$.

Proof. The proof of this lemma follows on the lines of a result in (Kapoor, 1972), so we omit the details. \square

A function f , analytic in unit disc is said to be admissible if its lower q -order satisfies (2.12).

Theorem 2.3. Let $f \in H(\overline{D}; R)$ having the index- q , q -order $\rho_R(q)$ and lower q -order $\beta_R(q)$. Further let $\varphi(n) = |E_{\lambda_n}/E_{\lambda_{n+1}}|^{1/(\lambda_{n+1}-\lambda_n)}$ forms a nondecreasing function of n for $n > n_0$. Then

$$\beta_R(q) + A(q) \leq \liminf_{n \rightarrow \infty} \frac{\log^{[q-1]} \lambda_n}{\log \lambda_n - \log^+ \log^+ E_{\lambda_n}^p(f)(R/d)^{\lambda_n}}. \tag{2.13}$$

Proof. Using (2.8), we get

$$\beta_R(q) = \liminf_{r \rightarrow R} \frac{\log^{[q]} \overline{M}(r, f)}{\log(Rr/(R-r))} \leq \liminf_{r \rightarrow R} \frac{\log^{[q]} M(((\lambda r + (1-\lambda)R)/R))}{\log(Rr/(R-r))} = \beta_0(q).$$

\square

It can be easily seen that $G(s)$ satisfies the hypothesis of Lemma 4. Applying Lemma 2.1 for $\beta_0(q)$, it gives

$$\beta_R(q) + A(q) \leq \liminf_{n \rightarrow \infty} \frac{\log^{[q-1]} \lambda_n}{\log \lambda_n - \log^+ \log^+ E_{\lambda_n}^p(f)(R/d)^{\lambda_n}} \quad q = 2, 3, \dots,$$

combining Theorem 2.2 and 2.3 we get the following theorem:

Theorem 2.4. Let $f \in H(\overline{D}; R)$ having the index- q , q -order $\rho_R(q)$ and lower q -order $\beta_R(q)$. Further, let $\varphi(n) = |a_n/a_{n+1}|^{1/(\lambda_{n+1}-\lambda_n)}$ forms a nondecreasing function of n for $n > n_0$. Then

$$\beta_R(q) + A(q) = \liminf_{n \rightarrow \infty} \frac{\log^{[q-1]} \lambda_{n-1}}{\log \lambda_n - \log^+ \log^+ E_{\lambda_n}^p(f)(R/d)^{\lambda_n}}. \tag{2.14}$$

Theorem 2.5. Let $f \in H(\bar{D}; R)$ having the index $-q$, q -order $\rho_R(q)$ and lower q -order $\beta_R(q)$. Then

$$\beta_R(q) + A(q) = \max_{\{n_k\}} \left[\liminf_{k \rightarrow \infty} \frac{\log^{[q-1]} \lambda_{n_{k-1}}}{\log \lambda_{n_k} - \log^+ \log^+ E_{\lambda_{n_k}}^P(f)(R/d)^{\lambda_{n_k}}} \right], \quad (2.15)$$

where maximum in (2.15) is taken overall increasing sequences $\{n_k\}$ of natural numbers.

Proof. Let $S(s) = \sum_{k=0}^{\infty} E_{\lambda_{n_k}}^P(f)(R/d)^{\lambda_{n_k}} s^{\lambda_{n_k}}$, $|s| < 1$, where $\{\lambda_{n_k}\}_{k=0}^{\infty}$ is the sequence of elements in the range set of $\nu(r)$. It can be easily seen that $G(s)$ and $S(s)$ have the same maximum term. Hence, the q -order and lower q -order of $S(s)$ are the same as those of $G(s)$. Thus, $S(s)$ is of lower q -order $\beta_R(q)$. Further, let $\xi(n_k) = \max\{r : \nu(r) = \lambda_{n_k}\}$. Then, $\xi(n_k) = \varphi(n_k)$, and consequently, $\varphi(n_k)$ is an increasing function of k . Therefore, $S(s)$ satisfies the hypothesis of Theorem 2.4 and so by (2.15) we get

$$\beta_R(q) + A(q) = \liminf_{k \rightarrow \infty} \frac{\log^{[q-1]} \lambda_{n_{k-1}}}{\log \lambda_{n_k} - \log^+ \log^+ E_{\lambda_{n_k}}^P(f)(R/d)^{\lambda_{n_k}}}. \quad (2.16)$$

But from Theorem 2.3, we get

$$\beta_R(q) + A(q) \geq \max_{\{n_k\}} \left[\liminf_{k \rightarrow \infty} \frac{\log^{[q-1]} \lambda_{n_{k-1}}}{\log \lambda_{n_k} - \log^+ \log^+ E_{\lambda_{n_k}}^P(f)(R/d)^{\lambda_{n_k}}} \right]. \quad (2.17)$$

Combining (2.16) and (2.17) we get (2.15). Hence the proof is complete. \square

References

- Giroux, A. (1980). Approximation of entire functions over bounded domains. *Journal of Approximation Theory* **28**(1), 45–53.
- Kapoor, G. P. (1972). On the lower order of functions analytic in the unit disc. *Math. Japon.* **17**(1), 45–54.
- Kapoor, G.P. and A. Nautiyal (1982). Approximation of entire functions over carathodory domains. *Bulletin of the Australian Mathematical Society* **25**, 221–229.
- Kapoor, G.P. and K. Gopal (1979). On the coefficients of functions analytic in the unit disc having fast rates of growth. *Annali di Matematica Pura ed Applicata* **121**(1), 337–349.
- Kumar, D. (2004). Coefficients characterization for functions analytic in the polydisc with fast growth. *Math. Sci. Res. J.* **8**(4), 128–136.
- Kumar, D. (2007a). Necessary conditions for L^p - convergence of Lagrange interpolation in finite disc. *International Journal of Pure and Applied Mathematics* **40**(2), 153–164.
- Kumar, D. (2007b). On approximation and interpolation errors of an analytic functions. *Fasc. Math.* **38**, 17–36.
- Kumar, D. (2010). On the fast growth of analytic functions by means of Lagrange polynomial approximation and interpolation in C^N . *Fasc. Math.* **13**, 85–99.
- Kumar, D. (2011). Growth and weighted polynomial approximation of analytic functions. *Transylvanian Journal of Mathematics and Mechanics* **3**(1), 23–30.
- Kumar, D. (2013). Slow growth and optimal approximation of pseudoanalytic functions on the disc. *International Journal of Analysis and Applications* **2**(1), 26–37.
- Kumar, D. and M. Amit (2006). On the growth of coefficients of analytic functions. *Math. Sci. Res. J.* **10**(11), 286–295.
- Markushevich, A. I. (1967). *Theory of Functions of a Complex Variable, Vol.III. Revised English Edition. (Translated and Edited by Richard A. Silverman. Prentice-Hall, Englewood Cliffs, New Jersey.*



A Mixed Integer Linear Programming Formulation for Restrained Roman Domination Problem

Marija Ivanović^a

^a*Faculty of Mathematics, University of Belgrade, Studentski trg 16/IV, 11 000 Belgrade, Serbia*

Abstract

This paper deals with a subgroup of Roman domination problems (RDP) named Restrained Roman domination problem (RRDP). It introduces a new mixed integer linear programming (MILP) formulation for the RRDP. The presented model uses relatively small number of the variables and constraints and could be of use both in theoretical and practical purposes. Proof of its correctness is given, i.e. it was shown that optimal solution to the RRDP formulation is equal to the optimal solution of the original problem.

Keywords: Restrained Roman domination in graphs, combinatorial optimization, integer linear programming.
2010 MSC: 90C11, 05C69.

1. Introduction

With contiguous territories throughout Europe, North Africa, and the Middle East, the Roman Empire was one of the largest in history (Kelly, 2006). The idea of building "empire without end" (Nicolet, 1991) expressed the ideology that neither time nor space limited the Empire. During the fourth century A.D., Emperor of Rome, Constantine the Great, intended to accomplish that idea. In order to expand the Roman Empire, he dealt with the next problem: How to organize legions such that entire Empire of Rome stayed defended? Since legions were highly trained, it was assumed that they could move fast from one city to another. City was considered to be defended if at least one legion was stationed in it or it was adjacent to a city with two legions within. The second condition was made because legion could move from a stationed city only if such an act won't leave it undefended.

Inspired by this historical problem, a new subgroup of the domination problems, named Roman domination problem (RDP), was proposed by Stewart (1999). RDP can be described as a problem of finding the minimal number of legions such that entire Empire of Rome is defended.

Email address: marijai@math.rs (Marija Ivanović)

More details about the RDP can be found in (ReVelle & Rosing, 2000), (Currò, 2014), (Liedloff *et al.*, 2005) and (Xing *et al.*, 2006).

Restrained Roman domination problem (RRDP), previously introduced by Pushpam & Sampath (2015), is defined also as a problem of finding the minimal number of legions such that entire Empire of Rome is defended but the conditions are slightly changed. Again, a city is considered to be defended if at least one legion is stationed within. But, a city without legion within is consider to be defended if it is adjacent to at least one city with two legions within and to at least one undefended city.

The Roman domination problem and the Restrained Roman domination problem can be illustrated by a graph such that each city of the Empire of Rome is represented by a vertex and, for two connected cities, the corresponding vertices are set to be adjacent.

Assuming that five cities, marked by numbers 1 - 5, are constructed such that a city marked by 1 is only adjacent to a city marked by 2 and that all other cities are adjacent to each other, a small illustration of the RDP and RRDP solutions are given in the figure below.

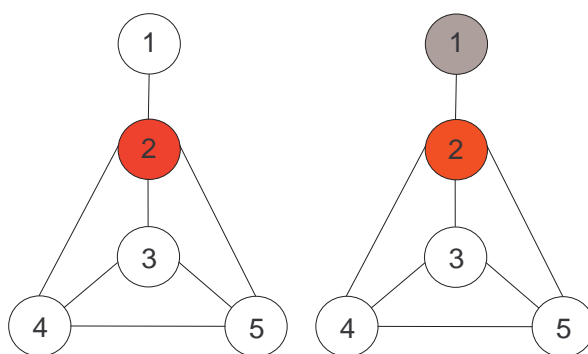


Figure 1. Illustrations of the the RDP (left) and RRDP (right) solutions

Vertex colored in red indicates that corresponding city is defended by two legions, vertex colored in gray indicates that corresponding city is defended by one legion, while vertices colored in white stands for the cities without legions within. For the Roman domination problem (shown on the figure on the left) by using only two legions, all five cities could be defended, i.e. assigning two legions to a city marked by 2, corresponding city and all adjacent cities are consider to be defended. Note that minimal number of legions for the RRDP (shown on the figure on the right) is three, i.e. assigning two legions to a city marked by 2, corresponding city and cities marked by 3, 4 and 5 are set to be defended because they are adjacent to a city with two legions within and adjacent to two cities with no assigned legions; city marked by 1 is set to be defended by one legion, since it can't be adjacent to at least one city without legions within and to at least one city with two legions within, at the same time. Given solution for RRDP is not unique since the same result could be obtained by assigning two legions to a city marked by 3 instead of the city marked by 2.

In the next sections, MILP formulation for the RRDP together with the proof of its validity, are proposed.

2. Problem definition

Let $G = (V, E)$ be an undirected graph with a vertex set V such that each vertex $u \in V$ represents a city of Roman Empire and each edge, $e \in E$, represents an existing road between two adjacent cities. A neighborhood set $N_u (N_u \subset V)$, of a vertex $u \in V$, is defined as a set of vertices v adjacent to a vertex u . For a function f

$$f : V \rightarrow \{0, 1, 2\} \tag{2.1}$$

let a number of legions assigned to a city represented by a vertex u to be equal to a value $f(u)$. Additionally, let a function f satisfy the condition that for every vertex $u \in V$ such that $f(u) = 0$ there exists vertices $v, w \in V$ such that $f(v) = 2$ and $f(w) = 0$. In other words, if there is an undefended city u , then there exist at least one city $v, v \in N_u$ with two legions within and at least one undefended city $w, w \in N_u$. Function f is called a restrained Roman domination function.

Mathematically, a proposed problem can be formulated as:

$$\min_f F_1(f) \tag{2.2}$$

subject to:

$$F_1(f) = \sum_{u \in V} f(u) \tag{2.3}$$

$$(\forall u \in V) f(u) = 0 \Rightarrow (\exists v, w \in N_u)(f(v) = 2 \wedge f(w) = 0). \tag{2.4}$$

Now, using a proposed notations, a solution to the illustrated RRDP can be written as: $F_1(f) = 3$ for $f(2) = 2, f(1) = 1$ and $f(3) = f(4) = f(5) = 0$ and it is not unique ($F_1(f) = 3$ for $f(3) = 2, f(1) = 1$ and $f(2) = f(4) = f(5) = 0$).

3. A mixed integer linear programming formulation for the RRDP

For a function f , defined by (2.1), let a continuous decision variable $x_i, x_i \in [0, \infty)$, indicate a number of legions assigned to a corresponding city $i \in V$. Although, $f \in \{0, 1, 2\}$ and $x_i \in [0, \infty)$, x_i and $f(i)$ are with equal values in the optimal solution, and not necessary with equal values for every feasible solution. Let binary decision variables y_i and z_i indicate if there are two or none legions assigned to a corresponding city $i \in V$,

$$y_i = \begin{cases} 1, & f(i) = 2 \\ 0, & \text{otherwise} \end{cases} \quad \text{and} \quad z_i = \begin{cases} 1, & f(i) = 0 \\ 0, & \text{otherwise} \end{cases} .$$

A mixed integer linear programming (MILP) formulation for the RRDP can now be formulated as follows:

$$\min \sum_{i \in V} x_i \tag{3.1}$$

subject to

$$x_i + \sum_{j \in N_i} y_j \geq 1, \quad i \in V \tag{3.2}$$

$$x_i + \sum_{j \in N_i} z_j \geq 1, \quad i \in V \tag{3.3}$$

$$x_i \geq 2y_i, \quad i \in V \tag{3.4}$$

$$x_i + 2z_i \leq 2, \quad i \in V \tag{3.5}$$

$$x_i \in [0, +\infty); \quad y_i, z_i \in \{0, 1\}, \quad i \in V. \tag{3.6}$$

Further, for vector values $x = [x_i], y = [y_i]$ and $z = [z_i]$, which satisfies constraints (3.2) - (3.6), notation $F_2(x, y, z) = \sum_{i \in V} x_i$ will be used. Now, condition (3.1) which minimizes the number of legions, can be written as $\min_{(x,y,z)} F_2(x, y, z)$. By the constraints (3.2) it is ensured that each undefended vertex i is adjacent to at least one vertex with two legions within. Similarly, by the constraints (3.3) it is ensured that each undefended vertex i is adjacent to at least one vertex which is also undefended. From the inequalities (3.4) and (3.5) it follows that for each city $i \in V$ with at most 1 legion within, corresponding value y_i is set to be equal to zero and that for each city $i \in V$ with at least one legions within, corresponding value z_i is set to be equal to zero. Finally, decision variables x are set to be continuous, while y and z are set to be binary by the constraints (3.6).

A given MILP formulation consists of $2|V|$ variables which are binary and $|V|$ continuous variables. Number of constraints is equal to $4|V|$.

A proof of the validity of the MILP formulation for the RRDP is given in the next proposition.

Proposition 1. *The optimal objective function value $F_1(f)$ of the Restrained Roman domination problem (2.1) - (2.4) is equal to the optimal objective function value $F_2(x, y, z)$ of the MILP formulation (3.1) - (3.6).*

Proof. (\Rightarrow) In this part will be proven that the optimal objective function value of the Restrained Roman domination problem (2.1) - (2.4) is greater or equal to the optimal objective function value of the MILP formulation (3.1) - (3.6), i.e. $F_1(f) \geq F_2(x, y, z)$.

For a fixed city $i \in V$ and a function f given by (2.1), let decision variables x_i, y_i and z_i be defined as

$$x_i = f(i), \quad y_i = \begin{cases} 1, & f(i) = 2 \\ 0, & \text{otherwise} \end{cases} \quad \text{and} \quad z_i = \begin{cases} 1, & f(i) = 0 \\ 0, & \text{otherwise} \end{cases}.$$

Since $x_i \in [0, +\infty), y_i, z_i \in \{0, 1\}$, conditions (3.4) - (3.6) are satisfied by the definition. For example, a condition (3.5) is satisfied because $z_i = 1$ for $x_i = f(i) = 0$ ($x_i + 2z_i = 2$) and $z_i = 0$ for

$x_i = f(i) = 1$ ($x_i + 2z_i = 1 < 2$). Similarly $z_i = 0$ for $x_i = f(i) = 2$, which again implies that $x_i + 2z_i = 2$.

Assuming that conditions (3.2) and (3.3) holds for a fixed vertex $i \in V$, there are two cases:

Case 1. Let values $f(i)$ are set to be greater or equal to one. Since $x_i = f(i)$, relation $x_i \geq 1$ implies. From the last relation and by the binary notations of the variables y_i and z_i it implies that $x_i + \sum_{j \in N_i} y_j \geq 1$ and $x_i + \sum_{j \in N_i} z_j \geq 1$.

Case 2. Let values $f(i)$ are set to be equal to zero. Satisfying relation (2.4) ($\exists v, w \in N_u$)($f(v) = 2 \wedge f(w) = 0$), it follows that $y_v = 1$ and $z_w = 1$. Therefore, $x_i + \sum_{j \in N_i} y_j = \sum_{j \in N_i} y_j \geq 1$ and $x_i + \sum_{j \in N_i} z_j = \sum_{j \in N_i} z_j \geq 1$.

Finally, since decision variables satisfies the conditions (3.1) - (3.6) for a fixed vertex i , it follows that $F_2(x, y, z) = \sum_{i \in V} x_i = \sum_{i \in V} f(i) = F_1(f)$.

(\Leftarrow) In this part it will be proven that optimal objective function value of the Restrained Roman domination problem (2.1) - (2.4) is less or equal to the optimal objective function value of the MILP formulation (3.1) - (3.6), i.e. $F_1(f) \leq F_2(x, y, z)$.

For a given set of decision variables x_i, y_i and z_i which satisfy conditions (3.1) - (3.6), let a function f be defined as

$$f(i) = \begin{cases} 0, & x_i \in [0, 1) \\ 1, & x_i \in [1, 2) \\ 2, & x_i \in [2, +\infty) \end{cases} \quad (3.7)$$

By the definition of the function f , condition (2.1) holds. Since the condition (2.1) holds, for a fixed vertex $u \in V$ there are two cases:

Case 1. Let $x_u \in [1, +\infty)$. By the definition of the function f , it follows that $f(u) = 1$ or $f(u) = 2$. Now, condition (2.4) holds, since $\perp \Rightarrow p$ is tautology for any logical statement p .

Case 2. Let $x_u \in [0, 1)$. By the definition of the function f , $f(u) = 0$. Because of the condition (3.2), $x_u + \sum_{j \in N_u} y_j \geq 1$, it follows that $\sum_{j \in N_u} y_j \geq 1 - x_u > 0$. Since the decision variables y_j are binary, $\sum_{j \in N_u} y_j$ has to be integer, which implies that $\sum_{j \in N_u} y_j \geq 1$. Therefore, there exists a vertex $v \in N_u$, $y_v = 1$. From the constraints (3.4), and because of the $x_v \geq 2y_v = 2$, it follows that $f(v) = 2$. Similarly, from the constraints (3.3) it follows that $\sum_{j \in N_u} z_j \geq 1 - x_u > 0$. Because of the binary type of the decision variables z_j , $\sum_{j \in N_u} z_j$ has an integer value. Now, since $\sum_{j \in N_u} z_j \geq 1$, there exists a vertex $w \in N_u$ such that $z_w = 1$. Finally, by the constraints (3.5), $x_w \leq 2 - 2z_w = 0$, it follows that $x_w = 0$ and that $f(w) = 0$ which means that condition (2.4) holds also.

By the definition of the function f , it is clear that $f(i) \leq x_i$, for $i \in V$. Therefore, $F_1(f) = \sum_{i \in V} f(i) \leq \sum_{i \in V} x_i = F_2(x, y, z)$.

So, for each feasible solution to the problem (2.1) - (2.4) there exists a feasible solution to the problem (3.1) - (3.6), satisfying the relation $F_2(x, y, z) \leq F_1(f)$, and for each feasible solution to the (3.1) - (3.6) there exists a feasible solution to the (2.1) - (2.4) satisfying the relation $F_1(f) \leq$

$F_2(x, y, z)$. Therefore, it follows that $\min_f F_1(f) = \min_{(x,y,z)} F_2(x, y, z)$. \square

Applying the given MILP formulation to the illustrated RRDP, solution $\min_{(x,y,z)} F_2(x, y, z)$ to the proposed problem is equal to 3, and it can be obtained for $x = [1, 0, 2, 0, 0]$, $y = [0, 1, 0, 0, 0]$ and $z = [0, 0, 1, 1, 1]$.

4. Conclusions

This paper is devoted to the Restrained Roman domination problem. A mixed integer linear programming formulation is introduced and the correctness of the corresponding formulation is proved. The presented model uses relatively small number of the variables and constraints, which indicates that presented model can be used both in theoretical and practical considerations. As a future study, it is planned to construct an exact method for solving the corresponding mathematical model. Construction of the metaheuristics for solving the proposed problem can also be a part of a possible future study.

Acknowledgments This research has been supported by the Research Grants 174010 and TR36015 of the Serbia Ministry of Education, Science and Technological Developments.

References

- Currò, Vincenzo (2014). The Roman Domination Problem on Grid Graphs. PhD thesis. Università di Catania.
- Kelly, Christopher (2006). *The Roman Empire: A Very Short Introduction*. Oxford University Press.
- Liedloff, Mathieu, Ton Kloks, Jiping Liu and Sheng-Lung Peng (2005). Roman domination over some graph classes. In: *Graph-Theoretic Concepts in Computer Science*. Springer. pp. 103–114.
- Nicolet, Claude (1991). *Space, Geography, and Politics in the Early Roman Empire*. University of Michigan Press.
- Pushpam, Roushini Leely and Padmapriya Sampath (2015). Restrained roman domination in graphs. *Transactions on Combinatorics* **4**(1), 1–17.
- ReVelle, Charles S and Kenneth E Rosing (2000). Defendens imperium romanum: a classical problem in military strategy. *American Mathematical Monthly* pp. 585–594.
- Stewart, Ian (1999). Defend the roman empire!. *Scientific American* **281**, 136–138.
- Xing, Hua-Ming, Xin Chen and Xue-Gang Chen (2006). A note on roman domination in graphs. *Discrete mathematics* **306**(24), 3338–3340.



New Čebyšev Type Inequalities for Functions whose Second Derivatives are (s_1, m_1) - (s_2, m_2) -convex on the Co-ordinates

B. Meftah^a, K. Boukerrioua^{a,*}

^aUniversity of Guelma. Guelma, Algeria.

Abstract

In this paper, we establish some new Čebyšev type inequalities for functions whose second derivatives are (s_1, m_1) - (s_2, m_2) -convex on the co-ordinates.

Keywords: Čebyšev type inequalities, co-ordinates (s_1, m_1) - (s_2, m_2) -convex, integral inequality.
2010 MSC: 26D15, 26D20, 39A12.

1. Introduction

In 1882, Čebyšev ([Chebyshev, 1882](#)) gave the following inequality

$$|T(f, g)| \leq \frac{1}{12} (b - a)^2 \|f'\|_{\infty} \|g'\|_{\infty}, \quad (1.1)$$

where $f, g : [a, b] \rightarrow \mathbb{R}$ are absolutely continuous functions, whose first derivatives f' and g' are bounded, where

$$T(f, g) = \frac{1}{b-a} \int_a^b f(x) g(x) dx - \left(\frac{1}{b-a} \int_a^b f(x) dx \right) \left(\frac{1}{b-a} \int_a^b g(x) dx \right), \quad (1.2)$$

and $\|\cdot\|_{\infty}$ denotes the norm in $L_{\infty}[a, b]$ defined as $\|f\|_{\infty} = \text{ess sup}_{t \in [a, b]} |f(t)|$.

During the past few years, many researchers established various generalizations, extensions and variants of Čebyšev type inequalities, we can mention the works ([Ahmad et al., 2009](#); [Boukerrioua & Guezane-Lakoud, 2007](#); [Guazene-Lakoud & Aissaoui, 2011](#); [Latif & Alomari, 2009](#);

*Corresponding author

Email address: khaledv2004@yahoo.fr (K. Boukerrioua)

Pachpatte & Talkies, 2006; Pachpatte, 2006; Sarikaya et al., 2014). Recently the authors of (Guazene-Lakoud & Aissaoui, 2011), established a new Čebyšev type inequality for functions of two independent variables whose second derivatives are bounded. Also in (Sarikaya et al., 2014), the authors obtained some new Čebyšev type inequalities involving functions whose mixed partial derivatives are s -convex on the co-ordinates. The main purpose of this work is to obtain new Čebyšev type inequalities for functions whose mixed partial derivatives are (s_1, m_1) - (s_2, m_2) -convex on the co-ordinates.

This paper is organized as follows: In section 2, we present some preliminaries. In the third section, we prove a new identity for functions of two independent variables then we used it to establish new Čebyšev type inequalities for functions whose mixed partial derivatives are (s_1, m_1) - (s_2, m_2) -convex on the co-ordinates.

2. Preliminaries

Throughout this paper we denote by Δ the bidimensional interval in $[0, \infty)^2$, $\Delta =: [a, b] \times [c, d]$ with $a < b$ and $c < d$, $\Delta_0 =: [0, b^*] \times [0, d^*]$ with $b^* > b$, $d^* > d$, $k =: (b - a)(d - c)$ and $\frac{\partial^2 f}{\partial \lambda \partial \alpha}$ by $f_{\lambda\alpha}$.

Definition 2.1. (Dragomir, 2001) A function $f : \Delta \rightarrow \mathbb{R}$ is said to be convex on the co-ordinates on Δ , if the following inequality:

$$f(\lambda x + (1 - \lambda)t, \alpha y + (1 - \alpha)v) \leq \lambda \alpha f(x, y) + \lambda(1 - \alpha)f(x, v) + (1 - \lambda)\alpha f(t, y) + (1 - \lambda)(1 - \alpha)f(t, v), \quad (2.1)$$

holds for all $\lambda, \alpha \in [0, 1]$ and $(x, y), (x, v), (t, y), (t, v) \in \Delta$.

Clearly, every convex mapping $f : \Delta \rightarrow \mathbb{R}$ is convex on the co-ordinates. Furthermore, it exists a co-ordinated convex function which is not convex.

Definition 2.2. (Alomari & Darus, 2008) A function $f : \Delta \rightarrow \mathbb{R}$ is said to be s -convex in the second sense on the co-ordinates on Δ , if the following inequality:

$$f(\lambda x + (1 - \lambda)t, \alpha y + (1 - \alpha)v) \leq \lambda^s \alpha^s f(x, y) + \lambda^s (1 - \alpha)^s f(x, v) + (1 - \lambda)^s \alpha^s f(t, y) + (1 - \lambda)^s (1 - \alpha)^s f(t, v), \quad (2.2)$$

holds for all $\lambda, \alpha \in [0, 1]$ and $(x, y), (x, v), (t, y), (t, v) \in \Delta$, for some fixed $s \in (0, 1]$.

s -convexity on the co-ordinates does not imply the s -convexity, it exist a functions which are s -convex on the co-ordinates but are not s -convex.

Definition 2.3. (Bai & Qi, 2013; Chun, 2014) A function $f : \Delta_0 \rightarrow \mathbb{R}$ is said (s, m) -convex on Δ , if the following inequality

$$f(\lambda x + m(1 - \lambda)t, \lambda y + m(1 - \lambda)v) \leq \lambda^s f(x, y) + m(1 - \lambda^s)f(t, v), \quad (2.3)$$

holds for all $(x, y), (t, v) \in \Delta$ and $\lambda \in [0, 1]$ and for some fixed $s, m \in (0, 1]$.

Definition 2.4. (Bai & Qi, 2013; Chun, 2014) A function $f : \Delta_0 \rightarrow \mathbb{R}$ is said to be (s_1, m_1) - (s_2, m_2) -convex on the co-ordinates on Δ_0 , if the following inequality

$$\begin{aligned}
 f(\lambda x + m_1(1 - \lambda)t, \alpha y + m_2(1 - \alpha)v) \leq & \lambda^{s_1} \alpha^{s_2} f(x, y) + m_2 \lambda^{s_1} (1 - \alpha^{s_2}) f(x, v) \\
 & + m_1 (1 - \lambda^{s_1}) \alpha^{s_2} f(t, y) \\
 & + m_1 m_2 (1 - \lambda^{s_1}) (1 - \alpha^{s_2}) f(t, v), \tag{2.4}
 \end{aligned}$$

holds for all $(x, y), (x, v), (t, y), (t, v) \in \Delta$ with $\lambda, \alpha \in [0, 1]$ and $s_1, m_1, s_2, m_2 \in (0, 1]$.

3. Main result

Lemma 3.1. Let $f : \Delta \rightarrow \mathbb{R}$ be partially differentiable function on Δ in \mathbb{R}^2 . If $f_{\lambda\alpha} \in L_1(\Delta)$, then for any $(x, y) \in \Delta \subset \Delta_0$, we have the following identity

$$\begin{aligned}
 f(x, y) = & \frac{1}{(b-a)} \int_a^b f(m_1 t, y) dt + \frac{1}{(d-c)} \int_c^d f(x, m_2 z) dz \\
 & - \frac{1}{k} \int_a^b \int_c^d f(m_1 t, m_2 z) dz dt + \frac{1}{k} \int_a^b \int_c^d (x - m_1 t)(y - m_2 z) \\
 & \times \left(\int_0^1 \int_0^1 f_{\lambda\alpha}(\lambda x + m_1(1 - \lambda)t, \alpha y - m_2(1 - \alpha)z) d\alpha d\lambda \right) dz dt, \tag{3.1}
 \end{aligned}$$

where $k = (b - a)(d - c)$.

Proof. For any $x, t \in [m_1 a, m_1 b]$ and $y, z \in [m_2 c, m_2 d]$ such that $t \neq x, y \neq z$, we have

$$\begin{aligned}
 \int_{m_1 t}^x \int_{m_2 z}^y f_{\sigma\tau}(\sigma, \tau) d\tau d\sigma &= \int_{m_1 t}^x (f_\sigma(\sigma, y) - f_\sigma(\sigma, m_2 z)) d\sigma \\
 &= f(x, y) - f(x, m_2 z) - f(m_1 t, y) + f(m_1 t, m_2 z), \tag{3.2}
 \end{aligned}$$

which implies

$$f(x, y) = f(x, m_2 z) + f(m_1 t, y) - f(m_1 t, m_2 z) + \int_{m_1 t}^x \int_{m_2 z}^y f_{\sigma\tau}(\sigma, \tau) d\tau d\sigma. \tag{3.3}$$

For $\sigma = \lambda x + m_1(1 - \lambda)t$ and $\tau = \alpha y - m_2(1 - \alpha)z$, (3.3) becomes

$$\begin{aligned}
 f(x, y) = & f(x, m_2 z) + f(m_1 t, y) - f(m_1 t, m_2 z) \\
 & + (x - m_1 t)(y - m_2 z) \int_0^1 \int_0^1 f_{\lambda\alpha}(\lambda x + m_1(1 - \lambda)t, \alpha y - m_2(1 - \alpha)z) d\tau d\sigma. \tag{3.4}
 \end{aligned}$$

Integrating (3.4) over $[a, b] \times [c, d] \subset \Delta_0$, with respect to t, z , multiplying the resultant equality by $\frac{1}{k}$, we obtain the desired equality. □

Theorem 3.1. Let $f, g : \Delta_0 \rightarrow \mathbb{R}$ be partially differentiable functions such that their second derivatives $f_{\lambda\alpha}$ and $g_{\lambda\alpha}$ are integrable on Δ_0 , if $|f_{\lambda\alpha}|$ and $|g_{\lambda\alpha}|$ are (s_1, m_1) - (s_2, m_2) -convex on the co-ordinates, then we have

$$|T(f, g)| \leq \frac{(1 + m_1 s_1)(1 + m_2 s_2)}{8(m_1 m_2 k)^2(1 + s_1)(1 + s_2)} \int_{m_1 a}^{m_1 b} \int_{m_2 c}^{m_2 d} [M|g(x, y)| + N|f(x, y)|] \times [(x - m_1 a)^2 + (m_1 b - x)^2][(y - m_2 c)^2 + (m_2 d - y)^2] dy dx, \quad (3.5)$$

where

$$T(f, g) = \frac{1}{m_1 m_2 k} \int_{m_1 a}^{m_1 b} \int_{m_2 c}^{m_2 d} f(x, y) g(x, y) dy dx - \frac{(d - c)}{m_1^2 m_2 k^2} \int_{m_1 a}^{m_1 b} \int_{m_2 c}^{m_2 d} g(x, y) \left(\int_{m_1 a}^{m_1 b} f(t, y) dt \right) dy dx - \frac{(b - a)}{m_1 m_2^2 k^2} \int_{m_1 a}^{m_1 b} \int_{m_2 c}^{m_2 d} g(x, y) \left(\int_{m_2 c}^{m_2 d} f(x, z) dz \right) dy dx + \frac{1}{m_1^2 m_2^2 k^2} \left(\int_{m_1 a}^{m_1 b} \int_{m_2 c}^{m_2 d} f(x, y) dy dx \right) \left(\int_{m_1 a}^{m_1 b} \int_{m_2 c}^{m_2 d} g(t, z) dz dt \right), \quad (3.6)$$

$$M = \operatorname{ess\,sup}_{x, t \in [a, b], y, z \in [c, d]} [|f_{\lambda\alpha}(x, y)| + |f_{\lambda\alpha}(x, z)| + |f_{\lambda\alpha}(t, y)| + |f_{\lambda\alpha}(t, z)|],$$

$$N = \operatorname{ess\,sup}_{x, t \in [a, b], y, z \in [c, d]} [|g_{\lambda\alpha}(x, y)| + |g_{\lambda\alpha}(x, z)| + |g_{\lambda\alpha}(t, y)| + |g_{\lambda\alpha}(t, z)|],$$

$$(s_1, m_1), (s_2, m_2) \in (0, 1]^2 \text{ and } k = (b - a)(d - c).$$

Proof. By Lemma 3.1, we have

$$f(x, y) - \frac{1}{(b - a)} \int_a^b f(m_1 t, y) dt - \frac{1}{(d - c)} \int_c^d f(x, m_2 z) dz + \frac{1}{k} \int_a^b \int_c^d f(m_1 t, m_2 z) dz dt = \frac{1}{k} \int_a^b \int_c^d (x - m_1 t)(y - m_2 z) \left(\int_0^1 \int_0^1 f_{\lambda\alpha}(\lambda x + m_1(1 - \lambda)t, \alpha y - m_2(1 - \alpha)z) d\alpha d\lambda \right) dz dt, \quad (3.7)$$

and

$$g(x, y) - \frac{1}{(b - a)} \int_a^b g(m_1 t, y) dt - \frac{1}{(d - c)} \int_c^d g(x, m_2 z) dz + \frac{1}{k} \int_a^b \int_c^d g(m_1 t, m_2 z) dz dt = \frac{1}{k} \int_a^b \int_c^d (x - m_1 t)(y - m_2 z) \left(\int_0^1 \int_0^1 g_{\lambda\alpha}(\lambda x + m_1(1 - \lambda)t, \alpha y - m_2(1 - \alpha)z) d\alpha d\lambda \right) dz dt. \quad (3.8)$$

Multiplying (3.7) by $\frac{1}{2m_1m_2k} g(x, y)$ and (3.8) by $\frac{1}{2m_1m_2k} f(x, y)$, summing the resultant equalities, then integrating on $[m_1a, m_1b] \times [m_2c, m_2d]$ with respect to x, y , we get

$$\begin{aligned}
 & \frac{1}{m_1m_2k} \int_{m_1am_2c}^{m_1bm_2d} \int_{m_1am_2c}^{m_1bm_2d} g(x, y)f(x, y)dydx - \frac{(d-c)}{2m_1m_2k^2} \int_{m_1am_2c}^{m_1bm_2d} \int_{m_1am_2c}^{m_1bm_2d} g(x, y) \\
 & \times \left(\int_a^b f(m_1t, y)dt \right) dydx - \frac{(b-a)}{2m_1m_2k^2} \int_{m_1am_2c}^{m_1bm_2d} \int_{m_1am_2c}^{m_1bm_2d} g(x, y) \left(\int_c^d f(x, m_2z)dz \right) dydx \\
 & - \frac{(d-c)}{2m_1m_2k^2} \int_{m_1am_2c}^{m_1bm_2d} \int_{m_1am_2c}^{m_1bm_2d} f(x, y) \left(\int_a^b g(m_1t, y)dt \right) dydx \\
 & - \frac{(b-a)}{2m_1m_2k^2} \int_{m_1am_2c}^{m_1bm_2d} \int_{m_1am_2c}^{m_1bm_2d} f(x, y) \left(\int_c^d g(x, m_2z)dz \right) dydx \\
 & + \frac{1}{2m_1m_2k^2} \int_{m_1am_2c}^{m_1bm_2d} \int_{m_1am_2c}^{m_1bm_2d} g(x, y) \left(\int_a^b \int_c^d f(m_1t, m_2z)dzdt \right) dydx \\
 & + \frac{1}{2m_1m_2k^2} \int_{m_1am_2c}^{m_1bm_2d} \int_{m_1am_2c}^{m_1bm_2d} f(x, y) \left(\int_a^b \int_c^d g(m_1t, m_2z)dzdt \right) dydx \\
 = & \frac{1}{2m_1m_2k^2} \int_{m_1am_2c}^{m_1bm_2d} \int_{m_1am_2c}^{m_1bm_2d} g(x, y) \left[\int_a^b \int_c^d (x - m_1t) (y - m_2z) \right. \\
 & \times \left. \left(\int_0^1 \int_0^1 f_{\lambda\alpha} (\lambda x + m_1(1 - \lambda)t, \alpha y - m_2(1 - \alpha)z) d\alpha d\lambda \right) dzdt \right] dydx \\
 & + \frac{1}{2m_1m_2k^2} \int_{m_1am_2c}^{m_1bm_2d} \int_{m_1am_2c}^{m_1bm_2d} f(x, y) \left[\int_a^b \int_c^d (x - m_1t) (y - m_2z) \right. \\
 & \times \left. \left(\int_0^1 \int_0^1 g_{\lambda\alpha} (\lambda x + m_1(1 - \lambda)t, \alpha y - m_2(1 - \alpha)z) d\alpha d\lambda \right) dzdt \right] dydx. \tag{3.9}
 \end{aligned}$$

By Fubini’s Theorem, we obtain

$$\begin{aligned}
 T(f, g) &= \frac{1}{2m_1m_2k^2} \int \int_{m_1am_2c}^{m_1bm_2d} g(x, y) \left[\int_a^b \int_c^d (x - m_1t)(y - m_2z) \right. \\
 &\quad \times \left. \left(\int_0^1 \int_0^1 f_{\lambda\alpha}(\lambda x + m_1(1 - \lambda)t, \alpha y - m_2(1 - \alpha)z) d\alpha d\lambda \right) dzdt \right] dydx \\
 &\quad + \frac{1}{2m_1m_2k^2} \int \int_{m_1am_2c}^{m_1bm_2d} f(x, y) \left[\int_a^b \int_c^d (x - m_1t)(y - m_2z) \right. \\
 &\quad \times \left. \left(\int_0^1 \int_0^1 g_{\lambda\alpha}(\lambda x + m_1(1 - \lambda)t, \alpha y - m_2(1 - \alpha)z) d\alpha d\lambda \right) dzdt \right] dydx. \tag{3.10}
 \end{aligned}$$

Using the (s_1, m_1) - (s_2, m_2) -convexity and modulus, (3.10) gives

$$\begin{aligned}
 |T(f, g)| &\leq \frac{1}{2m_1m_2k^2} \int \int_{m_1am_2c}^{m_1bm_2d} |g(x, y)| \left[\int_a^b \int_c^d |x - m_1t||y - m_2z| \right. \\
 &\quad \times \left[\int_0^1 \int_0^1 \left(\lambda^{s_1}\alpha^{s_2} |f_{\lambda\alpha}(x, y)| + m_2\lambda^{s_1}(1 - \alpha^{s_2}) |f_{\lambda\alpha}(x, z)| + m_1(1 - \lambda^{s_1})\alpha^{s_2} |f_{\lambda\alpha}(t, y)| \right. \right. \\
 &\quad \left. \left. + m_1m_2(1 - \lambda^{s_1})(1 - \alpha^{s_2}) |f_{\lambda\alpha}(t, z)| \right) d\alpha d\lambda \right] dzdt \Big] dydx \\
 &\quad + \frac{1}{2m_1m_2k^2} \int \int_{m_1am_2c}^{m_1bm_2d} |f(x, y)| \left[\int_a^b \int_c^d |x - m_1t||y - m_2z| \right. \\
 &\quad \times \left[\int_0^1 \int_0^1 \left(\lambda^{s_1}\alpha^{s_2} |g_{\lambda\alpha}(x, y)| + m_2\lambda^{s_1}(1 - \alpha^{s_2}) |g_{\lambda\alpha}(x, z)| + m_1(1 - \lambda^{s_1})\alpha^{s_2} |g_{\lambda\alpha}(t, y)| \right. \right. \\
 &\quad \left. \left. + m_1m_2(1 - \lambda^{s_1})(1 - \alpha^{s_2}) |g_{\lambda\alpha}(t, z)| \right) d\alpha d\lambda \right] dzdt \Big] dydx. \tag{3.11}
 \end{aligned}$$

By a simple calculation, we have

$$\begin{aligned}
 |T(f, g)| &\leq \frac{(1 + m_1s_1)(1 + m_2s_2)M}{2m_1m_2k^2(1 + s_1)(1 + s_2)} \int \int_{m_1am_2c}^{m_1bm_2d} |g(x, y)| \times \left[\int_a^b \int_c^d |x - m_1t||y - m_2z| dzdt \right] dydx \\
 &\quad + \frac{(1 + m_1s_1)(1 + m_2s_2)N}{2m_1m_2k^2(1 + s_1)(1 + s_2)} \int \int_{m_1am_2c}^{m_1bm_2d} |f(x, y)| \times \left[\int_a^b \int_c^d |x - m_1t||y - m_2z| dzdt \right] dydx. \tag{3.12}
 \end{aligned}$$

Noting that

$$\int_a^b |x - m_1 t| dt = \frac{1}{2m_1} [(x - m_1 a)^2 + (m_1 b - x)^2], \quad (3.13)$$

$$\int_c^d |y - m_2 z| dz = \frac{1}{2m_2} [(y - m_2 c)^2 + (m_2 d - y)^2]. \quad (3.14)$$

Combining (3.12), (3.13) and (3.14), we obtain the required inequality. \square

Corollary 3.1. *Let $f, g : \Delta_0 \rightarrow \mathbb{R}$ be partially differentiable functions such that their second derivatives $f_{\lambda\alpha}$ and $g_{\lambda\alpha}$, are integrable on Δ_0 . If $|f_{\lambda\alpha}|$ and $|g_{\lambda\alpha}|$ are (s_1, s_2) -convex on the co-ordinates, then we have*

$$|T(f, g)| \leq \frac{1}{8k^2} \int_a^b \int_c^d [M |g(x, y)| + N |f(x, y)|] [(x-a)^2 + (b-x)^2] \times [(y-c)^2 + (d-y)^2] dy dx, \quad (3.15)$$

where

$$\begin{aligned} T(f, g) &= \frac{1}{k} \int_a^b \int_c^d f(x, y) g(x, y) dy dx - \frac{(d-c)}{k^2} \int_a^b \int_c^d g(x, y) \left(\int_a^b f(t, y) dt \right) dy dx \\ &\quad - \frac{(b-a)}{k^2} \int_a^b \int_c^d g(x, y) \left(\int_c^d f(x, z) dz \right) dy dx \\ &\quad + \frac{1}{k^2} \left(\int_a^b \int_c^d f(x, y) dy dx \right) \left(\int_a^b \int_c^d g(t, z) dz dt \right), \end{aligned} \quad (3.16)$$

M, N, k are defined as in Theorem 3.1 and $(s_1, s_2) \in (0, 1]^2$.

Proof. Applying Theorem 3.1, for $m_1 = m_2 = 1$, we obtain the desired inequality. \square

Corollary 3.2. *Under the same hypothesis of Theorem 3.1, if $|f_{\lambda\alpha}|$ and $|g_{\lambda\alpha}|$ are m_1 - m_2 -convex on the co-ordinates, then we have the following inequality*

$$\begin{aligned} |T(f, g)| &\leq \frac{(1+m_1)(1+m_2)}{32(m_1 m_2 k)^2} \int_{m_1 a m_2 c}^{m_1 b m_2 d} [M |g(x, y)| + N |f(x, y)|] \\ &\quad \times [(x - m_1 a)^2 + (m_1 b - x)^2] [(y - m_2 c)^2 + (m_2 d - y)^2] dy dx, \end{aligned} \quad (3.17)$$

where $T(f, g), M, N, k$ are defined as in Theorem 3.1 and $(m_1, m_2) \in (0, 1]^2$.

Proof. Using Theorem 3.1, for $s_1 = s_2 = 1$, we obtain the desired inequality. \square

Theorem 3.2. Under the same hypothesis of Theorem 3.1, we have the following inequality

$$|T(f, g)| \leq \frac{49}{3600} \left[\frac{(1 + m_1 s_1)(1 + m_2 s_2)}{(1 + s_1)(1 + s_2)} \right]^2 MNk^2 m_1^2 m_2^2, \tag{3.18}$$

where $T(f, g)$, M , N , (s_1, m_1) , (s_2, m_2) and k are defined as in Theorem 3.1.

Proof. Let F, G, \widetilde{F} and \widetilde{G} be defined as follows

$$F = f(x, y) - \frac{1}{(b-a)} \int_a^b f(m_1 t, y) dt - \frac{1}{(d-c)} \int_c^d f(x, m_2 z) dz + \frac{1}{k} \int_a^b \int_c^d f(m_1 t, m_2 z) dz dt,$$

$$G = g(x, y) - \frac{1}{(b-a)} \int_a^b g(m_1 t, y) dt - \frac{1}{(d-c)} \int_c^d g(x, m_2 z) dz + \frac{1}{k} \int_a^b \int_c^d g(m_1 t, m_2 z) dz dt,$$

$$\widetilde{F} = \frac{1}{k} \int_a^b \int_c^d (x - m_1 t)(y - m_2 z) \times \left(\int_0^1 \int_0^1 f_{\lambda\alpha}(\lambda x + m_1(1 - \lambda)t, \alpha y - m_2(1 - \alpha)z) d\alpha d\lambda \right) dz dt,$$

$$\widetilde{G} = \frac{1}{k} \int_a^b \int_c^d (x - m_1 t)(y - m_2 z) \times \left(\int_0^1 \int_0^1 g_{\lambda\alpha}(\lambda x + m_1(1 - \lambda)t, \alpha y - m_2(1 - \alpha)z) d\alpha d\lambda \right) dz dt.$$

By Lemma 3.1, we have

$$F = \widetilde{F} \text{ and } G = \widetilde{G}, \tag{3.19}$$

which implies

$$F \times G = \widetilde{F} \times \widetilde{G}. \tag{3.20}$$

Integrating both sides of (3.20) over $[m_1 a, m_1 b] \times [m_2 c, m_2 d]$ with respect to x, y , multiplying the resultant equality by $\frac{1}{m_1 m_2 k}$, using Fubini's Theorem and modulus, we get

$$\begin{aligned} |T(f, g)| \leq & \frac{1}{m_1 m_2 k^3} \int_{m_1 a}^{m_1 b} \int_{m_2 c}^{m_2 d} \left[\int_a^b \int_c^d |x - m_1 t| |y - m_2 z| \right. \\ & \times \left(\int_0^1 \int_0^1 |f_{\lambda\alpha}(\lambda x + m_1(1 - \lambda)t, \alpha y - m_2(1 - \alpha)z)| d\alpha d\lambda \right) dz dt \times \int_a^b \int_c^d |x - m_1 t| |y - m_2 z| \\ & \left. \times \left(\int_0^1 \int_0^1 |g_{\lambda\alpha}(\lambda x + m_1(1 - \lambda)t, \alpha y - m_2(1 - \alpha)z)| d\alpha d\lambda \right) dz dt \right] dy dx. \end{aligned} \tag{3.21}$$

Using the (s_1, m_1) - (s_2, m_2) -convexity, we obtain

$$\begin{aligned}
 |T(f, g)| &\leq \frac{1}{m_1 m_2 k^3} \int_{m_1 a}^{m_1 b} \int_{m_2 c}^{m_2 d} \left[\int_a^b \int_c^d |x - m_1 t| |y - m_2 z| \left[\int_0^1 \int_0^1 [\lambda^{s_1} \alpha^{s_2} |f_{\lambda\alpha}(x, y)| \right. \right. \\
 &\quad + m_2 \lambda^{s_1} (1 - \alpha^{s_2}) |f_{\lambda\alpha}(x, z)| + m_1 (1 - \lambda^{s_1}) \alpha^{s_2} |f_{\lambda\alpha}(t, y)| \\
 &\quad \left. \left. + m_1 m_2 (1 - \lambda^{s_1}) (1 - \alpha^{s_2}) |f_{\lambda\alpha}(t, z)| \right] d\alpha d\lambda \right] dz dt \\
 &\quad \times \left[\int_a^b \int_c^d |x - m_1 t| |y - m_2 z| \left[\int_0^1 \int_0^1 [\lambda^{s_1} \alpha^{s_2} |g_{\lambda\alpha}(x, y)| \right. \right. \\
 &\quad + m_2 \lambda^{s_1} (1 - \alpha^{s_2}) |g_{\lambda\alpha}(x, z)| + m_1 (1 - \lambda^{s_1}) \alpha^{s_2} |g_{\lambda\alpha}(t, y)| \\
 &\quad \left. \left. + m_1 m_2 (1 - \lambda^{s_1}) (1 - \alpha^{s_2}) |g_{\lambda\alpha}(t, z)| \right] d\alpha d\lambda \right] dz dt \Big] dy dx \\
 &\leq \left[\frac{(1 + m_1 s_1)(1 + m_2 s_2)}{(1 + s_1)(1 + s_2)} \right]^2 \frac{M \times N}{m_1 m_2 k^3} \\
 &\quad \times \left[\int_{m_1 a}^{m_1 b} \left[\int_a^b |x - m_1 t| dt \right]^2 dx \right] \left[\int_{m_2 c}^{m_2 d} \left[\int_c^d |y - m_2 z| dz \right]^2 dy \right]. \tag{3.22}
 \end{aligned}$$

Taking into account that

$$\left[\int_{m_1 a}^{m_1 b} \left[\int_a^b |x - m_1 t| dt \right]^2 dx \right] = \frac{7}{60} m_1^3 (b - a)^5 \tag{3.23}$$

and

$$\left[\int_{m_2 c}^{m_2 d} \left[\int_c^d |y - m_2 z| dz \right]^2 dy \right] = \frac{7}{60} m_2^3 (d - c)^5. \tag{3.24}$$

The desired inequality, will be obtained by combining (3.22), (3.23) and (3.24). □

Corollary 3.3. Let $f, g : \Delta_0 \rightarrow \mathbb{R}$ be partially differentiable functions such that their second derivatives $f_{\lambda\alpha}$ and $g_{\lambda\alpha}$, are integrable on Δ_0 . If $|f_{\lambda\alpha}|$ and $|g_{\lambda\alpha}|$ are (s_1, s_2) -convex on the co-ordinates, then we have

$$|T(f, g)| \leq \frac{49}{3600} M \times N \times k^2, \tag{3.25}$$

where $T(f, g)$, M , N and k are defined as in Theorem 3.1.

Proof. Applying Theorem 3.2, for $m_1 = m_2 = 1$, we obtain the desired inequality. □

Corollary 3.4. Under the same hypothesis of Theorem 3.1, if $|f_{\lambda\alpha}|$ and $|g_{\lambda\alpha}|$ are m_1 - m_2 -convex on the co-ordinates, we have the following inequality

$$|T(f, g)| \leq \frac{49}{57600} [(1 + m_1)(1 + m_2)]^2 M \times N \times k^2 m_1^2 \times m_2^2, \tag{3.26}$$

where $T(f, g)$, M , N and k are defined as in Theorem 3.1 and $m_1, m_2 \in (0, 1]$.

Proof. Applying Theorem 3.2, for $s_1 = s_2 = 1$, we obtain the desired inequality. □

4. Acknowledgements

The author would like to thank the anonymous referee for his/her valuable suggestions. This work has been supported by CNEPRU–MESRS–B01120120103 project grants.

References

- Ahmad, F., N. S. Barnett and S. S. Dragomir (2009). New weighted Ostrowski and Čebyšev type inequalities. *Non-linear Analysis: Theory, Methods & Applications* **71**(12), e1408–e1412.
- Alomari, M. and M. Darus (2008). The Hadamard's inequality for s-convex function of 2-variables on the co-ordinates. *International Journal of Math. Analysis* **2**(13), 629–638.
- Bai, S. P. and F. Qi (2013). Some inequalities for (s_1, m_1) - (s_2, m_2) -convex functions on the co-ordinates. *Global Journal of Mathematical Analysis* **1**(1), 22–28.
- Boukerrioua, K. and A Guezane-Lakoud (2007). On generalization of Čebyšev type inequalities. *J. Inequal. Pure Appl. Math* **8**(2), Paper No. 55, 4 p.
- Chebyshev, P. L. (1882). Sur les expressions approximatives des integrales definies par les autres prises entre les mêmes limites. *In Proc. Math. Soc. Charkov* (Vol. **2**), 93–98.
- Chun, L. (2014). Some new inequalities of Hermite-Hadamard type for (α_1, m_1) - (α_2, m_2) -convex functions on coordinates. *Journal of Function Spaces* **5950**, Article ID 975950, 7.
- Dragomir, S. S. (2001). On Hadamard's inequality for convex functions on the co-ordinates in a rectangle from the plane. *Taiwanese J Math.* **4**, 775–788.
- Guazene-Lakoud, A. and F. Aissaoui (2011). New Čebyšev type inequalities for double integrals. *J. Math. Inequal.* **5**(4), 453–462.
- Latif, M. A. and M. Alomari (2009). On Hadamard-type inequalities for h-convex functions on the co-ordinates. *International Journal of Math. Analysis* **3**(33), 1645–1656.
- Pachpatte, B. G. (2006). On Čebyšev-Grüss type inequalities via Pečarić's extension of the Montgomery identity. *JIPAM. Journal of Inequalities in Pure & Applied Mathematics [electronic only]* **7**(1), 1–4.
- Pachpatte, B. G. and N. A. Talkies (2006). On Čebyšev type inequalities involving functions whose derivatives belong to L_p spaces. *J. Inequal. Pure and Appl. Math* **7**(2), 1–6.
- Sarikaya, M. Z., H. Budak and H. Yaldiz (2014). Čebysev type inequalities for co-ordinated convex functions. *Pure and Applied Mathematics Letters* **2**, 244–48.



Hadamard Product of Certain Harmonic Univalent Meromorphic Functions

R. M. El-Ashwah^a, B. A. Frasin^{b,*}

^a*Department of Mathematics, Faculty of Science, Damietta University, New Damietta 34517, Egypt.*

^b*Department of Mathematics, Al al-Bayt University, Mafraq, Jordan Department of Mathematics, Al al-Bayt University, Mafraq, Jordan.*

Abstract

In this paper the authors extended certain results concerning the Hadamard product for two classes related to star-like and convex harmonic univalent meromorphic functions, results for each class are obtained. Relevant connections of the results obtained here with various known results for meromorphic univalent functions are indicated.

Keywords: Harmonic functions, meromorphic functions, univalent functions, sense-preserving.

2010 MSC: 30C45, 30C50.

1. Introduction and definitions

A continuous function $f = u + iv$ is a complex valued harmonic function in a simply connected complex domain $D \subset \mathbb{C}$ if both u and v are real harmonic in D . It was shown by Clunie and Sheil-Small (Clunie & Sheil-Small, 1984) that such harmonic function can be represented by $f = h + \bar{g}$, where h and g are analytic in D . We call h the analytic part and g the co-analytic of f . Also, a necessary and sufficient condition for f to be locally univalent and sense-preserving in D is that $|h'(z)| > |g'(z)|$. There are numerous papers on univalent harmonic functions defined in a domain $U = \{z \in \mathbb{C} : |z| < 1\}$ (see (Jahangiri, 1998, 1999), and (Silverman & Silvia, 1999; Silverman, 1998)). Hengartner and Schober (Hengartner & Schober, 1987) investigated functions harmonic in the exterior of the unit disc, that is $U^* = \{z \in \mathbb{C} : |z| > 1\}$. They showed that a complex valued, harmonic, sense-preserving univalent function f , defined on U^* and satisfying $f(\infty) = \infty$ must admit the representation

$$f(z) = h(z) + \overline{g(z)} + A \log |z| \quad (A \in \mathbb{C}), \quad (1.1)$$

*Corresponding author

Email addresses: r_elashwah@yahoo.com (R. M. El-Ashwah), bafrasin@yahoo.com (B. A. Frasin)

where

$$h(z) = \alpha z + \sum_{n=1}^{\infty} a_n z^{-n} \quad , \quad g(z) = \beta z + \sum_{n=1}^{\infty} b_n z^{-n} \quad (0 \leq |\beta| < |\alpha|), \tag{1.2}$$

and $a = \overline{f_z}/f_z$ is analytic and satisfy $|a(z)| < 1$ for $z \in U^*$.

After this work, Jahangiri and Silverman (Jahangiri & Silverman, 1999) defined the class H_0^* of harmonic sense-preserving functions $f(z)$ that are starlike with respect to the origin in U^* given by (1.1) and (1.2) and proved that

$$\sum_{n=1}^{\infty} n(|a_n| + |b_n|) < |\alpha| - |\beta| - |A|.$$

Denote by Σ_H the class of meromorphic functions f that are harmonic univalent and sense-preserving in the exterior of open unit disc U in the form

$$f(z) = h(z) + \overline{g(z)} \tag{1.3}$$

where

$$h(z) = z + \sum_{n=1}^{\infty} a_n z^{-n} \quad , \quad g(z) = \sum_{n=1}^{\infty} b_n z^{-n}. \tag{1.4}$$

Also, Jahangiri (Jahangiri, 2002) proved that if $f(z)$ given by (1.3) and (1.4) belongs to $\Sigma_H^*(\gamma)$, then

$$\sum_{n=1}^{\infty} \left(\frac{n + \gamma}{1 - \gamma} |a_n| + \frac{n - \gamma}{1 - \gamma} |b_n| \right) < 1.$$

Several authors have studies the classes of harmonic univalent meromorphic functions (see (Ahuja & Jahangiri, 2002; El-Ashwah et al., 2014) and (Janteng & Halim, 2007)).

Now, we introduce the subclasses $\Sigma_H^*(c_n, d_n, \delta)$, $\Sigma_H^c(c_n, d_n, \delta)$ and $\Sigma_H^k(c_n, d_n, \delta)$ consisting of functions of the form (1.3) and (1.4) which satisfies the inequalities, respectively

$$\sum_{n=1}^{\infty} (c_n |a_n| + d_n |b_n|) < \delta \quad (c_n \geq d_n \geq c_2 > 0; \delta > 0), \tag{1.5}$$

$$\sum_{n=1}^{\infty} n (c_n |a_n| + d_n |b_n|) < \delta \quad (c_n \geq d_n \geq c_2 > 0; \delta > 0), \tag{1.6}$$

and

$$\sum_{n=1}^{\infty} n^k (c_n |a_n| + d_n |b_n|) < \delta \quad (c_n \geq d_n \geq c_2 > 0; \delta > 0). \tag{1.7}$$

It is easy to see that various subclasses of Σ_H consisting of functions $f(z)$ of the form (1.3) and (1.4) can be represented as $\Sigma_H^k(c_n, d_n, \delta)$ for suitable choices of c_n, d_n, δ and k studies earlier by various authors.

(i) $\Sigma_H^0(n, n, 1) = H_0^*$ (see Jahangiri and Silverman. ((Jahangiri & Silverman, 1999), with $\alpha = 1$ and $\beta = A = 0$));

(ii) $\Sigma_H^0(n + \gamma, n - \gamma, 1 - \gamma) = \Sigma_H^*(\gamma)(0 \leq \gamma < 1, n \geq 1)$ (see Jahangiri (Jahangiri, 2002));

(iii) $\Sigma_H^0(n(n+2)^m, n(n-2)^m, 1) = MH^*(m)(m \in \mathbb{N}_0 = \mathbb{N} \cup \{0\}, \mathbb{N} = \{1, 2, \dots\}, n \geq 1)$ (see Bostanci and Ozturk (Bostanci & Ozturk, 2010));

(vi) $\Sigma_H^0((n + \gamma)(n + 2)^m, (n - \gamma)(n - 2)^m, 1 - \gamma) = MH^*(m, \gamma)(0 \leq \gamma < 1, m \in \mathbb{N}_0, n \geq 1)$ (see Bostanci and Ozturk (Bostanci & Ozturk, 2011)).

Evidently, $\Sigma_H^0(c_n, d_n, \delta) = \Sigma_H^*(c_n, d_n, \delta)$, and $\Sigma_H^1(c_n, d_n, \delta) = \Sigma_H^c(c_n, d_n, \delta)$. Further $\Sigma_H^{k_1}(c_n, d_n, \delta) \subset \Sigma_H^{k_2}(c_n, d_n, \delta)$ if $k_1 > k_2 \geq 0$, the containment being proper. Moreover, for any positive integer k , we have the following inclusion relation

$$\Sigma_H^k(c_n, d_n, \delta) \subset \Sigma_H^{k-1}(c_n, d_n, \delta) \subset \dots \subset \Sigma_H^2(c_n, d_n, \delta) \subset \Sigma_H^c(c_n, d_n, \delta) \subset \Sigma_H^*(c_n, d_n, \delta).$$

We also note that for any nonnegative real number k , the class $\Sigma_H^k(c_n, d_n, \delta)$ is nonempty as the function

$$f(z) = z + \sum_{n=1}^{\infty} n^{-k} \frac{\delta}{c_n} \lambda_n z^{-n} + \sum_{n=1}^{\infty} n^{-k} \frac{\delta}{d_n} \lambda_n \overline{z^{-n}}$$

where $\lambda_n \geq 0$ and $\sum_{n=1}^{\infty} \lambda_n \leq 1$, satisfy the inequality (1.7).

For harmonic meromorphic functions of the form

$$f(z) = z + \sum_{n=1}^{\infty} |a_n| z^{-n} + \sum_{n=1}^{\infty} |b_n| \overline{z^{-n}}$$

and

$$G(z) = z + \sum_{n=1}^{\infty} A_n z^{-n} + \sum_{n=1}^{\infty} B_n \overline{z^{-n}} \quad (A_n, B_n \geq 0),$$

we define the convolution of two harmonic functions f and G as

$$\begin{aligned} (f * G)(z) &= f(z) * G(z) \\ &= z + \sum_{n=1}^{\infty} |a_n| A_n z^{-n} + \sum_{n=1}^{\infty} |b_n| B_n \overline{z^{-n}}, \end{aligned}$$

and similarly, we can define the convolution of more than two meromorphic functions.

Several authors such as Mogra (Mogra, 1994, 1991), Aouf and Darwish (Aouf & Darwish, 2006), El-Ashwah and Aouf (El-Ashwah & Aouf, 2009, 2011) studied the convolution properties of meromorphic functions only.

The object of this paper is to establish a results concerning the Hadamard product of functions in the classes $\Sigma_H^k(c_n, d_n, \delta)$, $\Sigma_H^c(c_n, d_n, \delta)$ and $\Sigma_H^*(c_n, d_n, \delta)$.

Throughout this paper, we assume $f(z)$, $g(z)$, $f_i(z)$, and $g_j(z)$ having following form

$$f(z) = z + \sum_{n=1}^{\infty} a_n z^{-n} + \sum_{n=1}^{\infty} \overline{b_n z^{-n}}, \tag{1.8}$$

$$g(z) = z + \sum_{n=1}^{\infty} u_n z^{-n} + \sum_{n=1}^{\infty} \overline{v_n z^{-n}}, \tag{1.9}$$

$$f_i(z) = z + \sum_{n=1}^{\infty} a_{n,i} z^{-n} + \sum_{n=1}^{\infty} \overline{b_{n,i} z^{-n}} \quad (i = 1, 2, \dots, s), \tag{1.10}$$

$$g_j(z) = z + \sum_{n=1}^{\infty} u_{n,j} z^{-n} + \sum_{n=1}^{\infty} \overline{v_{n,j} z^{-n}} \quad (j = 1, 2, \dots, q). \tag{1.11}$$

Since to a certain extent the work in the harmonic univalent meromorphic functions case has paralleled that of the harmonic analytic univalent case, one is tempted to search analogous results to those of Porwal and Dixt (Porwal & Dixit, 2015) for meromorphic harmonic univalent functions in U^* .

2. Main Results

Theorem 1. *Let the functions $f_i(z)$ defined by (1.10) belong to the class $\Sigma_H^c(c_n, d_n, \delta)$ for every $i = 1, 2, \dots, s$; and let the functions $g_j(z)$ defined by (1.11) belong to the class $\Sigma_H^*(c_n, d_n, \delta)$ for every $j = 1, 2, \dots, q$. If $c_n, d_n \geq n\delta, (n \geq 1)$, then the Hadamard product $f_1 * f_2 * \dots * f_m * g_1 * g_2 * \dots * g_q(z)$ belongs to the class $\Sigma_H^{2s+q-1}(c_n, d_n, \delta)$.*

Proof. It is sufficient to show that

$$\sum_{n=1}^{\infty} n^{2s+q-1} \left[c_n \left(\prod_{i=1}^s |a_{n,i}| \prod_{j=1}^q |u_{n,j}| \right) + d_n \left(\prod_{i=1}^s |b_{n,i}| \prod_{j=1}^q |v_{n,j}| \right) \right] \leq \delta$$

Since $f_i(z) \in \Sigma_H^c(c_n, d_n, \delta)$, we have

$$\sum_{n=1}^{\infty} n (c_n |a_{n,i}| + d_n |b_{n,i}|) \leq \delta, \tag{2.1}$$

for every $i = 1, 2, \dots, s$, and therefore,

$$nc_n |a_{n,i}| \leq \delta \quad \text{or} \quad |a_{n,i}| \leq \left(\frac{\delta}{nc_n} \right)$$

and hence

$$|a_{n,i}| \leq n^{-2}, \tag{2.2}$$

for every $i = 1, 2, \dots, s$. Also,

$$nd_n |b_{n,i}| \leq \delta \quad \text{or} \quad |b_{n,i}| \leq \left(\frac{\delta}{nd_n} \right)$$

and hence

$$|b_{n,i}| \leq n^{-2}, \tag{2.3}$$

for every $i = 1, 2, \dots, s$. Further, since $g_j(z) \in \Sigma_H^*(c_n, d_n, \delta)$, we have

$$\sum_{n=1}^{\infty} (c_n |u_{n,j}| + d_n |v_{n,j}|) \leq \delta, \quad (2.4)$$

for every $j = 1, 2, \dots, q$. Hence we obtain

$$|u_{n,j}| \leq n^{-1} \text{ and } |v_{n,j}| \leq n^{-1} \quad (2.5)$$

for every $j = 1, 2, \dots, q$.

Using (2.2) and (2.3) for $i = 1, 2, \dots, s$; (2.5) for $j = 1, 2, \dots, q-1$ and (2.4) for $j = q$, we have

$$\begin{aligned} & \sum_{n=1}^{\infty} n^{2s+q-1} \left[c_n \left(\prod_{i=1}^s |a_{n,i}| \prod_{j=1}^{q-1} |u_{n,j}| \right) |u_{n,q}| + d_n \left(\prod_{i=1}^s |b_{n,i}| \prod_{j=1}^{q-1} |v_{n,j}| \right) |v_{n,q}| \right] \\ & \leq \sum_{n=1}^{\infty} n^{2s+q-1} \left[c_n n^{-2s} n^{-(q-1)} |u_{n,q}| + d_n n^{-2s} n^{-(q-1)} |v_{n,q}| \right] \\ & = \sum_{n=1}^{\infty} c_n |u_{n,q}| + d_n |v_{n,q}| \leq \delta. \end{aligned}$$

Hence $f_1 * f_2 * \dots * f_m * g_1 * g_2 * \dots * g_q \in \Sigma_H^{2s+q-1}(c_n, d_n, \delta)$.

We note that the required estimate can also be obtained by using (2.2) and (2.3) for $i = 1, 2, \dots, s-1$; (2.5) for $j = 1, 2, \dots, q$; and (2.1) for $i = s$. \square

Taking into account the convolution of the functions $f_i(z)$ defined by (1.10) for every $i = 1, 2, \dots, s$; only in the proof of the above theorem and using (2.2) and (2.3) for $i = 1, 2, \dots, s-1$, and the relation (2.1) for $i = s$, we have the following corollary:

Corollary 1. Let the functions $f_i(z)$ defined by (1.10) belong to the class $\Sigma_H^c(c_n, d_n, \delta)$ for every $i = 1, 2, \dots, s$. If $c_n, d_n \geq n\delta$ ($n \geq 1$), then the Hadamard product $f_1 * f_2 * \dots * f_s(z)$ belongs to the class $\Sigma_H^{2s-1}(c_n, d_n, \delta)$.

Taking into account the convolution of the functions $g_j(z)$ defined by (1.11) for every $j = 1, 2, \dots, q$; only in the proof of the above theorem and using (2.5) for $j = 1, 2, \dots, q-1$; and the relation (2.4) for $j = q$, we have the following corollary:

Corollary 2. Let the functions $g_j(z)$ defined by (1.11) belong to the class $\Sigma_H^*(c_n, d_n, \delta)$ for every $j = 1, 2, \dots, q$. If $c_n, d_n \geq \delta$, ($n \geq 1$), then the Hadamard product $g_1 * g_2 * \dots * g_q$ belongs to the class $\Sigma_H^{q-1}(c_n, d_n, \delta)$.

Remarks (i) If the co-analytic parts of $f_i(z)$ and $g_j(z)$ are zero for every $i = 1, 2, \dots, s$ and $j = 1, 2, \dots, q$, then we obtain the results obtained by El-Ashwah and Aouf (El-Ashwah & Aouf, 2011), with $a_{0,i} = 1, i = 1, 2, \dots, s$ and $b_{0,j} = 1, j = 1, 2, \dots, q$;

(ii) Taking $c_n = n + 1 + \beta(n + 2\alpha - 1)$ and $\delta = 2\beta(1 - \alpha)$ ($0 \leq \alpha < 1, 0 < \beta \leq 1$) with the co-analytic parts zero in the above results, we obtain the results obtained by Mogra (Mogra, 1994);

(iii) Taking $c_n = n + \alpha$ and $\delta = 1 - \alpha$ ($0 \leq \alpha < 1$) with co-analytic parts are zero in the above results, we obtain the result obtained by Mogra (Mogra, 1991);

(iv) Taking $c_n = n^m[(n + 1) + \beta(n + 2\alpha - 1)]$ and $\delta = 2\beta(1 - \alpha)$ ($0 \leq \alpha < 1, 0 < \beta \leq 1, m \in \mathbb{N}_0$) with co-analytic parts are zero in the above results, we obtain the results obtained by El-Ashwah and Aouf (El-Ashwah & Aouf, 2009), with $a_{0,i} = 1, i = 1, 2, \dots, s$ and $b_{0,j} = 1, j = 1, 2, \dots, q$;

(vi) By specializing the coefficients c_n, d_n and the parameter δ we obtain corresponding results for various subclasses such as $H_0^*, \Sigma_H^*(\gamma), MH^*(m), MH^*(m, \gamma)$.

Acknowledgments

The authors thank the referees for their valuable suggestions which led to improvement of this study.

References

- Ahuja, O. P. and J. M. Jahangiri (2002). Certain meromorphic harmonic functions. *Bull. Malaysian Math. Sci. Soc* **25**, 1–10.
- Aouf, M. K. and H. E. Darwish (2006). Hadamard product of certain meromorphic univalent functions with positive coefficients. *South. Asian Bull. Math.* **30**, 23–28.
- Bostanci, H. and M. Ozturk (2010). A new subclass of the meromorphic harmonic starlike functions. *Appl. Math. Letters* **23**, 1027–1032.
- Bostanci, H. and M. Ozturk (2011). A new subclass of the meromorphic harmonic -starlike functions. *Appl. Math. Comput.* **218**, 683–688.
- Clunie, J. and T. Sheil-Small (1984). Harmonic univalent functions. *Ann. Acad. Sci. Fenn. Ser. A. I. Math.* **9**, 3–25.
- El-Ashwah, R.M. and M.K. Aouf (2009). Hadamard product of certain meromorphic starlike and convex functions. *Comput. Math. Appl.* **57**(7), 1102–1106.
- El-Ashwah, R.M. and M.K. Aouf (2011). The Hadamard product of meromorphic univalent functions defined by convolution. *Appl. Math. Letters* **24**, 1153–1157.
- El-Ashwah, R.M., J. Dziok M.K. Aouf and J. Stankiewicz (2014). Partial sums of certain harmonic univalent meromorphic functions. *J. Math. Anal.* **37**, 5–11.
- Henartner, W. and G. Schober (1987). Univalent harmonic function. *Trans. Amer. Math. Soc.* **299**(2), 1–31.
- Jahangiri, J. M. (1998). Coefficient bounds and univalent criteria for harmonic functions with negative coefficients. *Ann. Univ. Marie-Curie Sklodowska Sect. A* **52**, 57–66.
- Jahangiri, J. M. (1999). Harmonic functions starlike in the unit disc. *J. Math. Anal. Appl.* **235**(2), 470–477.
- Jahangiri, J. M. (2002). Harmonic meromorphic starlike functions. *Bull. Korean Math. Soc.* **37**(2), 291–301.
- Jahangiri, J. M. and H. Silverman (1999). Meromorphic univalent harmonic function with negative coefficients. *Bull. Korean Math. Soc.* **36**(4), 763–770.
- Janteng, A. and S. A. Halim (2007). A subclass of harmonic meromorphic functions. *Int. J. Contemp. Math. Sci.* **2**(24), 1167–1174.
- Mogra, M. L. (1991). Hadamard product of certain meromorphic univalent functions. *J. Math. Anal. Appl.* **157**(2), 10–16.
- Mogra, M. L. (1994). Hadamard product of certain meromorphic starlike and convex functions. *Tamkang J. Math.* **25**(2), 157–162.
- Porwal, S. and Dixit (2015). Convolution on a generalized class of harmonic meromorphic functions. *Kungpook Math. J.* **55**, 83–89.
- Silverman, H. (1998). Harmonic univalent function with negative coefficients. *J. Math. Anal. Appl.* **220**, 283–289.
- Silverman, H. and E. M. Silvia (1999). Subclasses of harmonic univalent functions. *New Zealand J. Math.* **28**, 275–284.



Katsaras's Type Fuzzy Norm under Triangular Norms

Sorin Nădăban^{a,*}, Tudor Bînzar^b, Flavius Pater^b, Carmen Țerei^a, Sorin Hoară^a

^a*Aurel Vlaicu University of Arad, Department of Mathematics and Computer Science, Elena Drăgoi 2, RO-310330 Arad, Romania.*

^b*"Politehnica" University of Timișoara, Department of Mathematics, Piața Victoriei 2, RO-300006 Timișoara, Romania.*

Abstract

The aim of this paper is to redefine Katsaras's fuzzy norm using the notion of t-norm.

Keywords: Fuzzy norm, fuzzy norm linear spaces, fuzzy subspaces, *-convexity.

2010 MSC: 46S40.

1. Introduction and preliminaries

The concept of fuzzy set was introduced by L.A. Zadeh in his famous paper (Zadeh, 1965). A fuzzy set in X is a function $\mu : X \rightarrow [0, 1]$. We will denote by $\mathcal{F}(X)$ the family of all fuzzy sets in X . The classical union and intersection of ordinary subsets of X can be extended by the following formulas, proposed by L. Zadeh:

$$\left(\bigvee_{i \in I} \mu_i \right) (x) = \sup \{ \mu_i(x) : i \in I \}, \quad \left(\bigwedge_{i \in I} \mu_i \right) (x) = \inf \{ \mu_i(x) : i \in I \}.$$

If $\mu_1, \mu_2 \in \mathcal{F}(X)$, then the inclusion $\mu_1 \subseteq \mu_2$ is defined by $\mu_1(x) \leq \mu_2(x)$.

Definition 1.1. (Chang, 1968) Let X, Y be arbitrary sets and $f : X \rightarrow Y$. If μ is a fuzzy set in Y , then $f^{-1}(\mu)$ is a fuzzy set in X defined by

$$f^{-1}(\mu)(x) = \mu(f(x)), (\forall)x \in X.$$

*Corresponding author

Email addresses: sorin.nadaban@uav.ro (Sorin Nădăban), tudor.binzar@upt.ro (Tudor Bînzar), flavius.pater@upt.ro (Flavius Pater), carmen.terei@uav.ro (Carmen Țerei), sorin.hoara@uav.ro (Sorin Hoară)

If μ is a fuzzy set in X then $f(\mu)$ is a fuzzy set in Y defined by

$$f(\mu)(y) = \begin{cases} \sup_{x \in f^{-1}(y)} \mu(x) & \text{if } f^{-1}(y) \neq \emptyset \\ 0 & \text{if } f^{-1}(y) = \emptyset \end{cases}.$$

Remark. Previous definition is a special case of Zadeh's extension principle.

Since then many authors have tried to investigate fuzzy sets and their applications from different points of view. An important problem was finding an adequate definition of a fuzzy normed linear space. In the studying of the fuzzy topological vector spaces, Katsaras (1984) introduced for the first time the notion of fuzzy norm on a linear space.

Definition 1.2. (Katsaras & Liu, 1977) A fuzzy set ρ in X is said to be:

1. convex if $t\rho + (1-t)\rho \subseteq \rho, (\forall)t \in [0, 1]$;
2. balanced if $\lambda\rho \subseteq \rho, (\forall)\lambda \in \mathbb{K}, |\lambda| \leq 1$;
3. absorbing if $\bigvee_{t>0} t\rho = 1$;
4. absolutely convex if it is both convex and balanced.

Proposition 1. (Katsaras & Liu, 1977) Let ρ be a fuzzy set in X . Then:

1. ρ is convex if and only if

$$\rho(tx + (1-t)y) \geq \rho(x) \wedge \rho(y), (\forall)x, y \in X, (\forall)t \in [0, 1];$$

2. ρ is balanced if and only if $\rho(\lambda x) \geq \rho(x), (\forall)x \in X, (\forall)\lambda \in \mathbb{K}, |\lambda| \leq 1$.

Definition 1.3. (Katsaras, 1984) A Katsaras fuzzy semi-norm on X is a fuzzy set ρ in X which is absolutely convex and absorbing.

Definition 1.4. (Nădăban & Dzitac, 2014) A fuzzy semi-norm ρ on X will be called Katsaras fuzzy norm if

$$\rho\left(\frac{x}{t}\right) = 1, (\forall)t > 0 \Rightarrow x = 0.$$

Remark. a) It is easy to see that

$$\left[\rho\left(\frac{x}{t}\right) = 1, (\forall)t > 0 \Rightarrow x = 0 \right] \Leftrightarrow \left[\inf_{t>0} \rho\left(\frac{x}{t}\right) < 1, \text{ for } x \neq 0 \right].$$

b) The condition $\left[\rho\left(\frac{x}{t}\right) = 1, (\forall)t > 0 \Rightarrow x = 0 \right]$ is much weaker than that imposed by Katsaras (1984),

$$\left[\inf_{t>0} \rho\left(\frac{x}{t}\right) = 0, \text{ for } x \neq 0 \right].$$

In 1992, Felbin (1992) introduced an idea of fuzzy norm on a linear space by assigning a fuzzy real number to each element of linear space. Following Cheng & Mordeson (1994), Bag & Samanta (2003) introduced another concept of fuzzy norm. In paper (Bag & Samanta, 2008) it is shown that the fuzzy norm defined by Bag and Samanta is similar to that of Katsaras. As the notion of fuzzy norm as defined by Cheng & Mordeson (1994) and Bag & Samanta (2003) can be generalized for arbitrary t-norms (see (Goleţ, 2010), (Alegre & Romaguera, 2010), (Nădăban & Dzitac, 2014)) motivates us to investigate the extension of Katsaras's fuzzy norm under triangular norm.

Definition 1.5. (Schweizer & Sklar, 1960) A binary operation

$$* : [0, 1] \times [0, 1] \rightarrow [0, 1]$$

is called triangular norm (t-norm) if it satisfies the following condition:

1. $a * b = b * a, (\forall) a, b \in [0, 1]$;
2. $a * 1 = a, (\forall) a \in [0, 1]$;
3. $(a * b) * c = a * (b * c), (\forall) a, b, c \in [0, 1]$;
4. If $a \leq c$ and $b \leq d$, with $a, b, c, d \in [0, 1]$, then $a * b \leq c * d$.

Example 1.1. Three basic examples of continuous t-norms are $\wedge, \cdot, *_L$, which are defined by $a \wedge b = \min\{a, b\}$, $a \cdot b = ab$ (usual multiplication in $[0, 1]$) and $a *_L b = \max\{a + b - 1, 0\}$ (the Lukasiewicz t-norm).

Remark. $a * 0 = 0, (\forall) a \in [0, 1]$.

Definition 1.6. (Nădăban & Dzitac, 2014) Let X be a vector space over a field \mathbb{K} and $*$ be a continuous t-norm. A fuzzy set N in $X \times [0, \infty)$ is called a Bag-Samanta's type fuzzy norm on X if it satisfies:

- (N1) $N(x, 0) = 0, (\forall) x \in X$;
- (N2) $[N(x, t) = 1, (\forall) t > 0]$ if and only if $x = 0$;
- (N3) $N(\lambda x, t) = N\left(x, \frac{t}{|\lambda|}\right), (\forall) x \in X, (\forall) t \geq 0, (\forall) \lambda \in \mathbb{K}^*$;
- (N4) $N(x + y, t + s) \geq N(x, t) * N(y, s), (\forall) x, y \in X, (\forall) t, s \geq 0$;
- (N5) $(\forall) x \in X, N(x, \cdot)$ is left continuous and $\lim_{t \rightarrow \infty} N(x, t) = 1$.

The triple $(X, N, *)$ will be called fuzzy normed linear space (briefly FNL-space).

Remark. Bag & Samanta (2003) gave this definition for $*$ = \wedge and Goleţ (2010), Alegre & Romaguera (2010) gave also this definition in the context of real vector spaces.

2. Fuzzy vector spaces under triangular norms

In paper (Das, 1988) the sum of fuzzy sets, fuzzy subspaces and convex fuzzy sets were re-defined using the notion of a t-norm. In this way, several results are obtained, some of which are generalisation of the results of Katsaras & Liu (1977).

Let X be a vector space over a field \mathbb{K} (where \mathbb{K} is \mathbb{R} or \mathbb{C}) and $*$ be a continuous t-norm.

Definition 2.1. Let μ_1, μ_2 be fuzzy sets in X . The sum of fuzzy sets μ_1, μ_2 is denoted by $\mu_1 + \mu_2$ and it is defined by

$$(\mu_1 + \mu_2)(x) = \sup_{x_1+x_2=x} [\mu_1(x_1) * \mu_2(x_2)] .$$

If μ is a fuzzy set in X and $\lambda \in \mathbb{K}$, then the fuzzy set $\lambda\mu$ is defined by

$$(\lambda\mu)(x) = \begin{cases} \mu\left(\frac{x}{\lambda}\right) & \text{if } \lambda \neq 0 \\ 0 & \text{if } \lambda = 0, x \neq 0 \\ \vee\{\mu(y) : y \in X\} & \text{if } \lambda = 0, x = 0 \end{cases} .$$

Remark. In the particular case in which $*$ = \wedge we obtain the definition introduced by Katsaras & Liu (1977).

Proposition 2. If $\alpha, \beta \in \mathbb{K}$ and $\mu, \mu_1, \mu_2 \in \mathcal{F}(X)$, then

1. $\alpha(\beta\mu) = \beta(\alpha\mu) = (\alpha\beta)\mu$;
2. $\mu_1 \leq \mu_2 \Rightarrow \alpha\mu_1 \leq \alpha\mu_2$.

Proof. 1) Case 1. $\alpha \neq 0, \beta \neq 0$.

$$(\alpha(\beta\mu))(x) = (\beta\mu)\left(\frac{x}{\alpha}\right) = \mu\left(\frac{x}{\alpha\beta}\right) = ((\alpha\beta)\mu)(x) .$$

Similarly,

$$(\beta(\alpha\mu))(x) = (\alpha\mu)\left(\frac{x}{\beta}\right) = \mu\left(\frac{x}{\alpha\beta}\right) = ((\alpha\beta)\mu)(x) .$$

Case 2. $\alpha = 0, \beta \neq 0$.

Let $x \neq 0$. Then

$$(\alpha(\beta\mu))(x) = 0; ((\alpha\beta)\mu)(x) = 0; (\beta(\alpha\mu))(x) = (\alpha\mu)\left(\frac{x}{\beta}\right) = 0 .$$

For $x = 0$ we have

$$\begin{aligned} (\alpha(\beta\mu))(x) &= \sup_{y \in X} (\beta\mu)(y) = \sup_{y \in X} \mu\left(\frac{y}{\beta}\right) = \sup_{y \in X} \mu(y) ; \\ ((\alpha\beta)\mu)(x) &= \sup_{y \in X} \mu(y) ; (\beta(\alpha\mu))(x) = (\alpha\mu)\left(\frac{x}{\beta}\right) = \sup_{y \in X} \mu(y) . \end{aligned}$$

Case 3. $\alpha \neq 0, \beta = 0$ is similar.

Case 4. $\alpha = 0, \beta = 0$.

For $x \neq 0$ we have

$$(\alpha(\beta\mu))(x) = 0; ((\alpha\beta)\mu)(x) = 0; (\beta(\alpha\mu))(x) = 0 .$$

If $x = 0$, then

$$\begin{aligned} (\alpha(\beta\mu))(x) &= \sup_{y \in X} (\beta\mu)(y) = \sup_{y \in X} \mu(y) . \\ ((\alpha\beta)\mu)(x) &= \sup_{y \in X} \mu(y) ; (\beta(\alpha\mu))(x) = \sup_{y \in X} (\alpha\mu)(y) = \sup_{y \in X} \mu(y) . \end{aligned}$$

2) Let $x \in X$.

Case 1. $\lambda \neq 0$.

$$(\lambda\mu_1)(x) = \mu_1\left(\frac{x}{\lambda}\right) \leq \mu_2\left(\frac{x}{\lambda}\right) = (\lambda\mu_2)(x) .$$

Case 2. $\lambda = 0$. If $x \neq 0$, then $(\lambda\mu_1)(x) = 0 = (\lambda\mu_2)(x)$. For $x = 0$, we have

$$(\lambda\mu_1)(x) = \sup_{y \in X} \mu_1(y) \leq \sup_{y \in X} \mu_2(y) = (\lambda\mu_2)(x)$$

□

Proposition 3. Let X, Y be vector spaces over \mathbb{K} , $f : X \rightarrow Y$ be a linear mapping, $\lambda \in \mathbb{K}$ and $\mu, \mu_1, \mu_2 \in \mathcal{F}(X)$. Then

1. $f(\mu_1 + \mu_2) = f(\mu_1) + f(\mu_2)$;
2. $f(\lambda\mu) = \lambda f(\mu)$.

Proof. The proof is exactly the same as in (Katsaras & Liu, 1977).

□

Proposition 4. Let $\mu, \mu_1, \mu_2 \in \mathcal{F}(X)$ and $\alpha, \beta \in \mathbb{K}$. The following sentences are equivalent:

1. $\alpha\mu_1 + \beta\mu_2 \leq \mu$;
2. For all $x, y \in X$ we have $\mu(\alpha x + \beta y) \geq \mu_1(x) * \mu_2(y)$.

Proof. The proof is exactly the same as in (Katsaras & Liu, 1977).

□

Definition 2.2. A fuzzy set μ in X is called *-fuzzy linear subspace of X if

1. $\mu + \mu \subseteq \mu$;
2. $\lambda\mu \subseteq \mu, (\forall)\lambda \in \mathbb{K}$.

Proposition 5. Let $\mu \in \mathcal{F}(X)$. Then μ is a *-fuzzy linear subspace of X if and only in

$$\mu(\alpha x + \beta y) \geq \mu(x) * \mu(y), (\forall)x, y \in X, (\forall)\alpha, \beta \in \mathbb{K} .$$

Proof. The proof is exactly the same as in (Katsaras & Liu, 1977).

□

Definition 2.3. A fuzzy set μ in X is called *-convex if

$$\mu(tx + (1 - t)y) \geq \mu(x) * \mu(y), (\forall)x, y \in X, (\forall)t \in [0, 1] .$$

Remark. If $\mu \in \mathcal{F}(X)$ is $*$ -convex and crisp, then $\mu = \phi_A$ (ϕ_A is the characteristic function of the subset A of X) and

$$\mu(tx + (1 - t)y) \geq \mu(x) * \mu(y), (\forall)x, y \in X, (\forall)t \in [0, 1].$$

Thus

$$\mu(tx + (1 - t)y) = 1, (\forall)x, y \in A, (\forall)t \in [0, 1].$$

Hence

$$tx + (1 - t)y \in A, (\forall)x, y \in A, (\forall)t \in [0, 1].$$

So A is convex in the classical sence.

Proposition 6. A fuzzy set μ in X is $*$ -convex if and only if

$$t\mu + (1 - t)\mu \subseteq \mu, (\forall)t \in [0, 1].$$

Proof. ” \Rightarrow ” Case 1. $t = 0$.

$$(0\mu + 1\mu)(x) = \sup_{x_1+x_2=x} [(0\mu)(x_1) * (1\mu)(x_2)] = \sup_{y \in X} \mu(y) * \mu(x) \leq 1 * \mu(x) = \mu(x).$$

Case 2. $t = 1$ is similar.

Case 3. $t \in (0, 1)$.

$$\begin{aligned} (t\mu + (1 - t)\mu)(x) &= \sup_{x_1+x_2=x} [(t\mu)(x_1) * ((1 - t)\mu)(x_2)] = \\ &= \sup_{x_1+x_2=x} \left[\mu\left(\frac{x_1}{t}\right) * \mu\left(\frac{x_2}{1 - t}\right) \right] \leq \\ &\leq \sup_{x_1+x_2=x} \mu \left[t \cdot \frac{x_1}{t} + (1 - t) \cdot \frac{x_2}{1 - t} \right] = \sup_{x_1+x_2=x} \mu(x_1 + x_2) = \mu(x). \end{aligned}$$

” \Leftarrow ” Case 1. $t \in (0, 1)$.

$$\mu(tx + (1 - t)y) \geq (t\mu + (1 - t)\mu)(tx + (1 - t)y) \geq (t\mu)(tx) * ((1 - t)\mu)((1 - t)y) = \mu(x) * \mu(y).$$

Case 2. $t = 0$.

$$\mu(0x + 1y) = \mu(y) = 1 * \mu(y) \geq \mu(x) * \mu(y).$$

Case 3. $t = 1$ is similar. □

Remark. Some $*$ -convexity properties of fuzzy sets, where $*$ is a triangular norm on $[0, 1]$, were investigated in papers (Yandong, 1984; Yuan & Lee, 2004; Nourouzi & Aghajani, 2008) etc.

3. Katsaras’s type fuzzy norm

Let X be a vector space over a field \mathbb{K} (where \mathbb{K} is \mathbb{R} or \mathbb{C}) and $*$ be a continuous t-norm.

Definition 3.1. A fuzzy set ρ in X which is $*$ -convex, balanced and absorbing will be called Katsaras’s type fuzzy semi-norm. If in addition

$$\rho\left(\frac{x}{t}\right) = 1, (\forall)t > 0 \Rightarrow x = 0,$$

then ρ will be called Katsaras’s type fuzzy norm.

Lemma 1. *If ρ is balanced and absorbing, then $\rho(0) = 1$.*

Proof. As ρ is balanced, we have that $\rho(0) = \rho(0 \cdot x) \geq \rho(x)$. Thus $\rho(0) = \bigvee_{x \in X} \rho(x)$. As ρ is absorbing, we have that $\bigvee_{t > 0} \rho\left(\frac{x}{t}\right) = 1$. Hence $\bigvee_{x \in X} \rho(x) = 1$. Thus $\rho(0) = 1$. □

Lemma 2. *If ρ is balanced, then $\rho(\alpha x) = \rho(|\alpha|x)$, $(\forall)x \in X, (\forall)\alpha \in \mathbb{K}$.*

Proof. If ρ is balanced, then $\rho(\lambda x) \geq \rho(x)$, $(\forall)x \in X, (\forall)\lambda \in \mathbb{K}, |\lambda| \leq 1$. Particularly, for $\lambda \in \mathbb{K} : |\lambda| = 1$, we have

$$\rho\left(\frac{1}{\lambda}x\right) \geq \rho(x), (\forall)x \in X.$$

Replacing x with λx , we obtain $\rho(x) \geq \rho(\lambda x)$, $(\forall)x \in X$. Thus

$$\rho(\lambda x) = \rho(x), (\forall)x \in X, (\forall)\lambda \in \mathbb{K}, |\lambda| = 1.$$

Take $\alpha \in \mathbb{K}, \alpha \neq 0$ (if $\alpha = 0$ it is obvious that $\rho(\alpha x) = \rho(|\alpha|x)$, $(\forall)x \in X$). We put in previous equality $\lambda = \frac{\alpha}{|\alpha|}$. It results

$$\rho\left(\frac{\alpha}{|\alpha|}x\right) = \rho(x), (\forall)x \in X \Leftrightarrow \rho(\alpha x) = \rho(|\alpha|x), (\forall)x \in X.$$

□

Remark. The following theorems extend some results obtained in (Bag & Samanta, 2008).

Theorem 1. *If ρ is a Katsaras’s type fuzzy norm, then*

$$N(x, t) := \begin{cases} \rho\left(\frac{x}{t}\right) & \text{if } t > 0 \\ 0 & \text{if } t = 0 \end{cases}$$

is a Bag-Samanta’s type fuzzy norm.

Proof. (N1) $N(x, 0) = 0, (\forall)x \in X$ is obvious.

(N2) $[N(x, t) = 1, (\forall)t > 0] \Rightarrow \rho\left(\frac{x}{t}\right) = 1, (\forall)t > 0 \Rightarrow x = 0$. Conversely, if $x = 0$, then $N(0, t) = \rho(0) = 1, (\forall)t > 0$.

(N3) We suppose that $t > 0$ (if $t = 0$ (N3) is obvious). Using previous lemma we have:

$$N(\lambda x, t) = \rho\left(\frac{\lambda x}{t}\right) = \rho\left(\frac{|\lambda|x}{t}\right) = \rho\left(\frac{x}{t/|\lambda|}\right) = N\left(x, \frac{t}{|\lambda|}\right).$$

(N4) If $t = 0$, then $N(x, t) = 0$ and $N(x, t) * N(y, s) = 0 * N(y, s) = 0$ and the inequality $N(x+y, t+s) \geq N(x, t) * N(y, s)$ is obvious. A similar situation occurs when $s = 0$. If $t > 0, s > 0$, then

$$N(x + y, t + s) = \rho\left(\frac{x + y}{t + s}\right) = \rho\left(\frac{t}{t + s} \cdot \frac{x}{t} + \frac{s}{t + s} \cdot \frac{y}{s}\right) \geq \rho\left(\frac{x}{t}\right) * \rho\left(\frac{y}{s}\right) = N(x, t) * N(y, s).$$

(N5) First, we note that $N(x, \cdot)$ is non-decreasing. Indeed, for $s > t$, we have

$$N(x, s) = N(x + 0, t + s - t) \geq N(x, t) * N(0, s - t) = N(x, t) * 1 = N(x, t).$$

We prove now that $N(x, \cdot)$ is left continuous in $t > 0$.

Case 1. $N(x, t) = 0$. Thus, for all $s \leq t$, as $N(x, s) \leq N(x, t)$, we have that $N(x, s) = 0$. Therefore $\lim_{s \rightarrow t, s < t} N(x, s) = 0 = N(x, t)$.

Case 2. $N(x, t) > 0$. Let α be arbitrary such that $0 < \alpha < N(x, t)$. Let (t_n) be a sequence such that $t_n \rightarrow t, t_n < t$. We prove that there exists $n_0 \in \mathbb{N}$ such that $N(x, t_n) \geq \alpha, (\forall)n \geq n_0$. As $\alpha \in (0, N(x, t))$ is arbitrary, we will obtain that $\lim_{n \rightarrow \infty} N(x, t_n) = N(x, t)$.

As $N(x, t) = t\rho(x) > 0$, we have that $\rho(x) > 0$. Let $s = \frac{\alpha}{\rho(x)}$. We note that $s < t$. Indeed,

$$s < t \Leftrightarrow \frac{\alpha}{\rho(x)} < t \Leftrightarrow \alpha < t\rho(x) = N(x, t).$$

As $t_n \rightarrow t, t_n < t$ and $s < t$, there exists $n_0 \in \mathbb{N}$ such that $t_n > s, (\forall)n \geq n_0$. Then

$$N(x, t_n) = \rho\left(\frac{x}{t_n}\right) \geq \rho\left(\frac{x}{s}\right) = s\rho(x) = \alpha.$$

Since $\bigvee_{t>0} t\rho(x) = 1$, we obtain that $\bigvee_{t>0} N(x, t) = 1$. Thus $\lim_{t \rightarrow \infty} N(x, t) = 1$. □

Theorem 2. *If N is a Bag-Samanta's type fuzzy norm, then $\rho : X \rightarrow [0, 1]$ defined by*

$$\rho(x) = N(x, 1), (\forall)x \in X$$

is a Katsaras's type fuzzy norm.

Proof. First, we note that, by (N2), we have $\rho(0) = N(0, 1) = 1$.

(1) ρ is *-convex.

Let $t \in (0, 1)$. Then

$$\rho(tx + (1 - t)y) = N(tx + (1 - t)y, 1) = N(tx + (1 - t)y, t + 1 - t) \geq$$

$$\geq N(tx, t) * N((1-t)y, 1-t) = N(x, 1) * N(y, 1) = \rho(x) * \rho(y) .$$

If $t = 0$, then $\rho(tx + (1-t)y) = \rho(y) = 1 * \rho(y) \geq \rho(x) * \rho(y)$. The case $t = 1$ is similar.

(2) ρ is balanced.

Let $x \in X, \lambda \in \mathbb{K}^*, |\lambda| \leq 1$. As $N(x, \cdot)$ is non-decreasing, we have that

$$\rho(\lambda x) = N(\lambda x, 1) = N\left(x, \frac{1}{|\lambda|}\right) \geq N(x, 1) = \rho(x) .$$

If $x \in X, \lambda = 0$, then $\rho(\lambda x) = \rho(0) = 1 \geq \rho(x)$.

(3) ρ is absorbing.

Using (N5), we have that

$$\bigvee_{t>0} (t\rho)(x) = \bigvee_{t>0} \rho\left(\frac{x}{t}\right) = \bigvee_{t>0} N\left(\frac{x}{t}, 1\right) = \bigvee_{t>0} N(x, t) = 1 .$$

Finally,

$$\rho\left(\frac{x}{t}\right) = 1, (\forall)t > 0 \Rightarrow N\left(\frac{x}{t}, 1\right) = 1, (\forall)t > 0 \Rightarrow N(x, t) = 1, (\forall)t > 0 \Rightarrow x = 0 .$$

□

Acknowledgments

This work was co-funded by European Union through European Regional Development Funds Structural Operational Program Increasing of Economic Competitiveness Priority axis 2, operation 2.1.2. Contract Number 621/2014.

References

- Alegre, C. and S. Romaguera (2010). Characterizations of fuzzy metrizable topological vector spaces and their asymmetric generalization in terms of fuzzy (quasi-)norms. *Fuzzy Sets and Systems* **161**, 2182–2192.
- Bag, T. and S.K. Samanta (2003). Finite dimensional fuzzy normed linear spaces. *Journal of Fuzzy Mathematics* **11**(3), 687–705.
- Bag, T. and S.K. Samanta (2008). A comparative study of fuzzy norms on a linear space. *Fuzzy Sets and Systems* **159**, 670–684.
- Chang, C.L. (1968). Fuzzy topological spaces. *J. Math. Anal. Appl.* **24**, 182–190.
- Cheng, S.C. and J.N. Mordeson (1994). Fuzzy linear operator and fuzzy normed linear spaces. *Bull. Calcutta Math. Soc.* **86**, 429–436.
- Das, P. (1988). Fuzzy vector spaces under triangular norms. *Fuzzy Sets and Systems* **25**(1), 73–85.
- Felbin, C. (1992). Finite dimensional fuzzy normed linear spaces. *Fuzzy Sets and Systems* **48**, 239–248.
- Goleř, I. (2010). On generalized fuzzy normed spaces and coincidence point theorems. *Fuzzy Sets and Systems* **161**, 1138–1144.
- Katsaras, A.K. (1984). Fuzzy topological vector spaces II. *Fuzzy Sets and Systems* **12**, 143–154.
- Katsaras, A.K. and D.B. Liu (1977). Fuzzy vector spaces and fuzzy topological vector spaces. *Journal of Mathematical Analysis and Applications* **58**, 135–146.

- Nourouzi, K. and A. Aghajani (2008). Convexity in triangular norm of fuzzy sets. *Chaos, Solitons and Fractals* **36**, 883–889.
- Nădăban, S. and I. Dzitac (2014). Atomic decompositions of fuzzy normed linear spaces for wavelet applications. *Informatica* **25**(4), 643–662.
- Schweizer, B. and A. Sklar (1960). Statistical metric spaces. *Pacific J. Math.* **10**, 314–334.
- Yandong, Yu (1984). On the convex fuzzy sets (I). *Fuzzy Mathematics* **3**, 29–39.
- Yuan, X.H. and E.S. Lee (2004). The definition of convex fuzzy subset. *Computers and Mathematics with Applications* **47**, 101–113.
- Zadeh, L.A. (1965). Fuzzy sets. *Informations and Control* **8**, 338–353.



Measure of Tessellation Quality of Voronoï Meshes

E. A-iyeh^a, J.F. Peters^{a,b,*}

^a*Computational Intelligence Laboratory, Department of Electrical & Computer Engineering,
University of Manitoba, Winnipeg, MB, R3T 5V6, Canada.*

^b*Department of Mathematics, Faculty of Arts and Sciences, Adiyaman University, Adiyaman, Turkey.*

Abstract

This article introduces a measure of the quality of Voronoï tessellations resulting from various mesh generators. Mathematical models of a number of mesh generators are given. A main result in this work is the identification of those mesh generators that produce the highest quality Voronoï tessellations. Examples illustrating the application of the quality measure are given in comparing Voronoï tessellations of digital images.

Keywords: Sites, Mesh Generation, Quality, Tessellations, Voronoï mesh.
2010 MSC: Primary 54E05, Secondary 20L05, 35B36.

1. Introduction

This article introduces a measure of the quality of Voronoï tessellations resulting from various mesh generators. A *Voronoï tessellation* is a collection of non-overlapping convex polygons called *Voronoï regions*. It is well-known that creating meshes is a fundamental and necessary step in several areas, including engineering, computing, geometric and scientific applications (Leibon & Letscher, 2000; Owen, 1998; Liu & Liu, 2004). Meshes assume simplex structures or volumes based on the geometry of the surfaces, dimension of the space and placement of sites of the meshes (see, e.g., (Ebeida & Mitchell, 2012; Mitchell, 1993; Persson, 2004)). This work is a natural outgrowth of recent work on Voronoï tessellation (Persson, 2004; Persson & Strang, 2004; Bern & Plassmann, 1999; Du *et al.*, 1999; Brauwerman *et al.*, 1999; Peters, 2015b).

Seeds, generating points or sites of meshes for non-image domains may be chosen randomly, deterministically on grids (Persson, 2004), using distribution sampling e.g. (Ebeida & Mitchell, 2012), using the centroids of tessellation regions. In the search for a measure of mesh quality, it is

*Corresponding author: 75A Chancellor's Circle, EITC-E2-390, University of Manitoba, WPG, MB R3T 5V6, Canada; e-mail: james.peters3@ad.umanitoba.ca, research supported by Natural Sciences & Engineering Research Council of Canada (NSERC) discovery grant 185986.

Email addresses: umaiyeh@myumanitoba.ca (E. A-iyeh), James.Peters3@umanitoba.ca (J.F. Peters)

useful to identify the best or most suitable mesh generators (also called sites) for mesh generation. A principal benefit of this work is the identification of those generators that produce the highest quality Voronoï tessellations.

A natural application of the proposed mesh quality measure is given in comparing and classifying digital images Voronoï tessellation covers. Each Voronoï region is a closed set of points in a convex polygon. A Voronoï tessellation *cover* of a digital image equals the union of the Voronoï regions.

2. Related Work

Numerous mesh generation algorithms have been developed for several purposes including image processing and segmentation (Arbeláez & Cohen, 2006), clustering (Ramella et al., 1998), data compression, quantization and territorial behavior of animals (Persson, 2004; Persson & Strang, 2004; Du et al., 1999). These methods tackle a wide variety of geometrical representations for meshing the surfaces. In addition, a number of mesh generation algorithms are iterative in nature, so that the algorithm adjusts the meshes iteratively to approach fulfilling predefined conditions, thereby terminating on meeting the criterion up to some limits (Persson, 2004; Persson & Strang, 2004). In some adaptive mesh algorithms (Persson, 2004; Persson & Strang, 2004; George, 2006), the mesh sites are variable. For example they may be displaced to attain force equilibrium. In one such scenario the algorithm starts by partitioning the space based on the initial distribution of sites but iterates through them according to preset conditions on an element size function until the equilibrium and stopping conditions are satisfied (Persson, 2004). This essentially is refining and solving for optimality of the meshes (Rajan, 1994; Peraire et al., 1987; Rivara, 1984; Ruppert, 1995) according to the preset conditions.

Voronoï diagrams introduced by the Ukrainian mathematician G. Voronoï (Voronoï, 1903, 1907, 1908) (elaborated in the context of proximity spaces in (Peters, 2015b), (Peters, 2015c), (Peters, 2015a)) provide a means of covering a space with a polygonal mesh. In telecommunications, Voronoï diagrams have furnished a tool of analysis for binary linear block codes (Agrell, 1996) governing regions of block code, performance of Gaussian channel among others. In music, Voronoï diagrams have once again demonstrated their utility (McLean, 2007). For example, they been successfully applied in automatic grouping of polyphony (Hamanaka & Hirata, 2002). Other works bordering on applications of Voronoï meshes in reservoir modeling (Møller & Skare, 2001), attempts at cancer diagnosis (Demir & Yener, 2005) are also available.

Ever since utility of Voronoï diagrams was demonstrated in several applications including the post office problem where given a set of post office sites in furnished the answer in determining the nearest one to visit and other few works of application of 56 meshed surfaces an area, Voronoï tessellation has been useful in the study of the territorial behavior of animals, image compression, segmentation etc no works or very few have focused on their application in proximity and classification analysis of digital images. This work proposes to explore the utility of meshes in the study of proximal regions as a means of image classification.

Mesh applications in image analysis are not widely studied in the open literature. In that regard, we seek to contribute to the application of meshes in image classification by (a) identifying the best forms of generating points for meshes, (b) arriving at a measure of the quality of meshes, and

(c) characterizing meshes best suited for image classification. In addition, it is anticipated that this research will yield useful theorems that are a natural outcome of measuring mesh quality. Such theorems provide a formal foundation for the study of image mesh quality.

3. Preliminaries

Since the discovery that Voronoï tessellations are the secret working formula of bees, humans have sought to bring similar benefits and applications of Voronoï tessellations in their applications to image processing, image compression, clustering, territorial behaviour of animals etc (Persson, 2004; Persson & Strang, 2004; Bern & Plassmann, 1999; Du et al., 1999; Brauwerman et al., 1999). Several forms of polygonal meshes exist due to Voronoï and Delaunay tessellations and since Delaunay triangulation is a dual of Voronoï, our focus will be on the former.

Assume a finite set S of locations called sites s_i in a space \mathbb{R}^n . Computing the Voronoï diagram of S means partitioning the space into Voronoï regions $V(s_i)$ in such a way that $V(s_i)$ contains all points of S that are closer to s_i than to any other object $s_j, i \neq j$ in S .

Given the generator set

$$S = \{s_1, \dots, s_k : i \in \mathbb{N}\},$$

where each member of S is called a mesh generating point, let $s_i \in S$. The Voronoï region $V(s_i)$ is defined by

$$V(s_i) = \{x \in \mathbb{R}^n : \|x - s_i\| \leq \|x - s_k\|, s_k \in S, i \neq k\},$$

where $\|.,.\|$ is the Euclidean norm (distance between vectors). The set

$$V(S) = \bigcup_{s_i \in S} V(s_i)$$

is called the n-dimensional Voronoï diagram generated by the points in S . In \mathbb{R}^2 , this effectively covers the plane with convex and non overlapping polygons, one for each generating point in S . A centroidal Voronoï tessellation is a special case of $V(s_i)$, where the sites are the mass centroids of regions computed by

$$c_i = \frac{\int_{V_i} x\rho(x)dx}{\int_{V_i} \rho(x)dx}$$

where $\rho(x)$ is the density function of $V(s_i)$. The mathematical utility of centroidal tessellations lie in their relationship to the energy function (Brauwerman et al., 1999) defined as:

$$E(z, V) = \sum_{i=1}^n \int_{V_i} \rho(x)|x - z_i|^2 dx.$$

The energy function $E(z, V)$ for a Voronoï region depends on the density function $\rho(x)$ of the regions and the squared distances between the generating site z_i and nearby points x in the region. The total energy for a Voronoï diagram is the sum of integrals of the individual energies of the regions comprising the Voronoï diagram.

Voronoï regions are cells of growth from a certain view point. This view point is equivalent to considering the vertex of each region as a nucleus of a growing or expanding cell. Cells propagate

simultaneously outward from their nuclei at uniform rates until they intersect with others. They then freeze giving the boundaries of the regions defined by the tessellation.

Ultimately, cells whose nuclei are on the convex hull of a vertex grow until they intersect the outgrowth of others. It is interesting to note that since all cells are growing at the same rate, their first points of contact will coincide with the midpoint of the two nuclei. This is exactly the locus of all equidistant points from the nuclei. In other words it is the perpendicular bisector of the line segment between the nuclei from which all points are equidistant. The set of all points on these loci form the edges of the regions.

Voronoi cells that share an edge are said to be Voronoi neighbors. The aggregate of triangles formed by connecting the nuclei of all Voronoi neighbors tessellate the area within the convex hull of the set. Notice that the regions obtained are neighborhoods defined by the norm $\| \cdot \|$. This makes it possible to have a continuous image-like treatment of a dot pattern, thus permitting the application of general image processing techniques (Ahuja & Schachter, 1982). One such interpretation of the Voronoi tessellation is to view it as a cluster partition of a space (Ramella et al., 1998). In this view, the space is segmented by the various Voronoi regions, with the size of the clusters given by the areas of the regions. Also, the Voronoi tessellation is useful for boundary extraction, with the tessellated space viewed as a mosaic.

Definition 3.1. Given a point set $S \subseteq \mathbb{R}^n$ and a distance function d_n , the set $\{V_i\}_{i=1}^k$ is called a Voronoi tessellation of S if $V_i \cap V_j \neq \emptyset$ for $i \neq j$.

Definition 3.2. The Voronoi region of a site is a polygon about that site. The set of all regions partition the plane of the sites S based on a distance function $\| \cdot \|$. This results in the plane being covered with polygons about those sites.

Definition 3.3. Given a set $S = \{s_1, \dots, s_k\}$ of points, any plane (v_i, v_j) is a Voronoi edge of the Voronoi region V if and only if there exists a point x such that the circle centered at x and circumscribing v_i and v_j does not contain in its interior any other point of V .

Definition 3.4. A Voronoi tessellation is a set of polygons with their edges and vertices that partition a given space of sites.

Definition 3.5. A Voronoi edge is a half plane equidistant from two sites and which bounds some part of a Voronoi region. Every edge is incident upon exactly two vertices and every vertex upon at least three edges.

Definition 3.6. A Voronoi vertex is the center of a circle through three sites.

Definition 3.7. A set of points form a convex set if there is a line connecting each pair of points.

Definition 3.8. The convex hull of Voronoi regions about a set of sites is the smallest set which contains the Voronoi regions as well as the union of the regions.

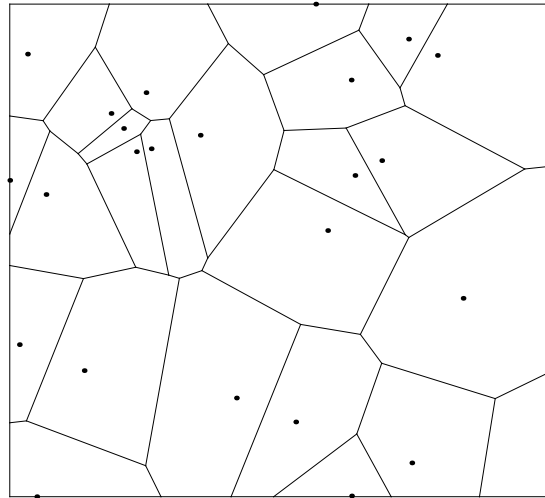


Figure 1. Voronoï diagram of a set of sites.

Definition 3.9. A dot or point pattern is a set of points of a signal representing spatial locations of signal features. For example sets of corners, keypoints etc are referred to as point or dot patterns.

Definition 3.10. The quality of a signal is defined as a characteristic of the signal which gives information about perceived signal degradation compared to an ideal.

Lemma 3.1. *The energy of a point located in a particular Voronoï region $V(s_i)$ is minimal with respect to all other regions $V(s_j)$ for $i \neq j$.*

Proof. Consider a point $y \in V(S_i)$. Its energy is evaluated as:

$$E(z, V) = \int_{V_i} \rho(y) |y - \hat{z}_i|^2 dy$$

$$E(z, V) = \int_{V_i} \rho(y) (0) dy = 0.$$

Since the distance between z and y is 0 by definition, $E = 0$. However $E(z, V)$ of $y \notin V(S_j)$ is

$$E(z, V) = \int_{V_j} \rho(y) |y - \hat{z}_i|^2 dy \neq 0.$$

Since $\rho(y)$ nor $|y - \hat{z}_i|^2$ is non-zero. □

Theorem 3.1. *The energy function $E(z, V)$ is minimized at the centroid sites of the tessellations.*

Proof.

$$E(z, V) = \sum_{i=1}^n \int_{V_i} \rho(x) |x - \hat{z}_i|^2 dx.$$

The energy of a Voronoï region V_i is the integral of the product of the density function of that region $\rho(x)$ and the squared distances between the generating site \hat{z}_i and points comprising the region. The total energy $E(z, V)$ is the sum of the energies of all Voronoï regions.

To obtain the minimum of $E(z, V)$, it requires that the derivative of the function with respect to the sites be equal to zero. The solution of the derivative are the sites \hat{z}_i .

$$\frac{dE}{d\hat{z}_i} = 2 \sum_{i=1}^n \int_{V_i} \rho(x)(x - \hat{z}_i)dx = 0$$

$$\hat{z}_i = \frac{\int_{V_i} x\rho(x)dx}{\int_{V_i} \rho(x)dx}.$$

The solution \hat{z}_i are the centroids. □

Theorem 3.2. For a given set of sites $Z = \{z_i\}$, the energy is minimized when V is a centroidal Voronoï tessellation.

Proof. Immediate from Lemma 3.1 and Theorem 3.1. □

4. Tessellation Generators

In the literature few works or none considers the choice of sites for meshing images using location of image features. We mostly go ahead and tessellate using chosen locations irrespective of locations of image features. Previously, sites have been chosen to correspond to center of masses of regions (Du et al., 1999; Burns, 2009), random locations (Ebeida et al., 2011; Aurenhammer, 1991), deterministic or regular locations (Persson, 2004; Persson & Strang, 2004; Aurenhammer, 1991). In short, majority of these locations do not take information of the sites into account. The method of centroids has evident advantages (Brauerman et al., 1999), but these are questionable if the regions from which we obtain the center of masses do not reflect image feature locations. In this work, various sites based on image features are first discovered and subsequently the sites give a tessellation of the space. With several sites found and tessellations performed, quality measures of the meshes are obtained with the view of helping ascertain the best sites for tessellations using an overall quality measure. This then forms a road map to meshed image analysis given a method of sites that yield high quality meshes. Important image features are known to reside at corner, edge, keypoints, centroids, extrema and modal image feature sites. These then would be used to discover mesh sites. They are treated next.

4.1. Image Corner Points

A very notable feature of digital images are image corner points. These define points where structures in the image intersect, as such they form a solid background for feature recognition and extraction.

One of the earliest criteria for corner point identification is a point that has low self-similarity (Moravec, 1980). Each pixel centered in a patch is compared to nearby pixels in an overlapping patch for corner point candidate examination. Since then, improvements in corner point identification by computing differential corner scores with respect to direction instead of patches in

(Moravec, 1980) resulted in combined corner and edge detection (Harris & Stephens, 1988). For more accuracy in subpixels in corner identification see (Förstner & Gülch, 1987). Recently, corner detection based on other methods: Multiple scales due originally to (Harris & Stephens, 1988), level curvature approach (Kitchen & Rosenfeld, 1980; Koenderink & Richards, 1988), difference of Laplacians, Gaussians, and Hessians (Lowe, 2004; Lindeberg, 1998, 2008), affine-adapted interest point operators (Lindeberg, 1993, 2008; Mikolajczyk & Schmid, 2004), curvature placement along edges (Wang & Brady, 1995), smallest univalue segment assimilating nucleus (Smith & Brady, 1997), direct testing of pixel self-similarity and feature accelerated segment (Rosten & Drummond, 2006), non-parametric and adaptive region processing methods (Guru & Dinesh, 2004), transform approaches (Kang et al., 2005; Park et al., 2004), adaptive approaches (Pan et al., 2014), structure-based analysis (Kim et al., 2012) have resulted. Corner points are identified here as points in which there is a significant change in intensity features in two or more directions:

$$C(u, v) = \sum w(x, y) [I(x + u, y + v) - I(x, y)]^2.$$

The window function $w(x, y)$ tends to be rectangular but could assume other suitable forms. So, a corner point is returned by the corner point function $C(x, y)$ if the squared intensity difference between the intensity at location (x, y) and location $(x + u, y + v)$ is large for any two directions.

4.2. Edge Sites

Edge maps are widely used in image processing for feature detection and object recognition. Besides, edge information is known to be crucial in feature detection and image analysis. These have been used due to the fact that the edges tend to localize an object of interest for target processing and feature detection. Edge points are points whose feature values differ sharply from those of neighboring points (Canny, 1986).

4.3. Image Keypoint Sites

Image keypoints are popular for extracting distinct image points (Lowe, 1999; Mitchell, 2010; Feng et al., 2013; Woźniak & Marszałek, 2014). Scale-space extrema detection has been shown to yield image keypoints (Lowe, 1999). This process however usually yields numerous keypoint candidates therefore we have to resort to means of reducing their number to important ones. For example by eliminating low contrast points, the most important ones are retained. Detection of locations of keypoints invariant to scale change may be accomplished by obtaining stable features across scales of an image (Lowe, 1999). In that regard, the scale space $L(x, y, \sigma)$ is obtained by convolution of Gaussian functions $G(x, y, \sigma)$ with the image function $I(x, y)$ at several scales k .

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y)$$

$$G(\vec{x}, \sigma) = \frac{1}{2\pi\sqrt{\sigma}} \text{Exp}\left\{-\frac{1}{2}(\vec{x} - \mu)^T \sigma^{-1}(\vec{x} - \mu)\right\}.$$

We proceed below to obtain the difference of the convolved result in the previous step.

$$D(x, y, \sigma) = (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y)$$

$$D(x, y, \sigma) = L(x, y, k\sigma) - L(x, y, \sigma).$$

In the final step, the extrema (minima and maxima) points are obtained by comparing points in the difference functions and selecting those that achieve the minimum or maximum values in their neighborhoods (Lowe, 1999). Keypoint locations originally found at $D(x, y)$ at scales of σ may be used as estimates in Taylor series expansion about the position vector $\vec{x} = (x, y)$ to obtain more accurate locations of the points (Brown & Lowe, 2002). Locations are extrapolated as follows.

$$D_a(\vec{x}) = D + \frac{\partial D^T}{\partial \vec{x}} + \frac{1}{2} \vec{x} \vec{x}^T \frac{\partial^2 D}{\partial \vec{x}^2 \vec{x}},$$

where $D_a(\vec{x})$ is the improved location of a keypoint.

4.4. Centroid Sites

Given a tessellated plane of points centroid points of regions are computed as follows.

$$c_i = \frac{\int_{V_i} x\rho(x)dx}{\int_{V_i} \rho(x)dx}.$$

The center of masses c_i s are computed from Voronoï regions and then used to re-tessellate the regions. Given corner, edge and keypoint sites in images, their tessellations produce Voronoï regions corresponding to those generators. The centers of masses of these regions based on those image feature locations form a set of sites in the plane for centroidal tessellations.

4.5. Modal Pixel and Extrema Sites

The histogram distributions of images are readily available. They furnish information on the distribution of the pixels in the image. Pivotal points in an image can be sought by considering the modal pixel positions in the image. In this way, we get to find out the feature value that occurs most frequently in a digital image and use the locations of those as sites for tessellations. A variant to this approach is to find the modal feature value, displace it by a constant and then use the positions of the resulting value as sites for image tessellations. The most influential feature M is obtained from the histogram distribution $h(k)$ from the steps below.

$$h(k) = \sum_x \sum_y n_k$$

$$M = \text{Max}_k(\sum_x \sum_y n_k) = \text{Max}_k(hk).$$

For a given image function $I(x, y)$, the feature values are in the range $[0, k]$. From this set, there exists minimum and maximum feature values. Sites corresponding to the extrema can be used for tessellations. However, it should be noted that where any of the extrema is unique, only one site is returned for the tessellation. Extrema sites M_1, M_2, M_d are useful because they can give us geometrical information about objects that tend to have a particular distribution or feature value;

$$M_1 = \text{Min}(f(x, y) \forall x, y)$$

$$M_2 = \text{Max}(f(x, y) \forall x, y)$$

$$M_d = \text{Max}(\sum \sum n_{ij}) - a.$$

Extrema sites are discovered using M_1 and M_2 , whilst displaced modal sites are discovered using M_d , given the constant a . Due to the numbers of modal and extrema sites and the nature of meshes they produce, they are not treated further in this work. Instead, displaced feature sites are discovered and used.

So far, we have identified sites of important features in digital images. A necessary step in determining the choice of sites lies in the quality of meshes produced by the particular choice of sites. In what follows, the quality of meshes produced by the sites is examined.

5. Voronoi Mesh Quality Analysis

Quality metrics for mesh analysis have been explored in the literature (Knupp, 2001; Shewchuk, 2002; Bhatia & Lawrence, 1990a). Most mesh generation approaches set a predefined quality factor for each cell as such they easily achieve meshes with high quality. This is not always useful or justifiable. For example, in images it's highly unlikely that mesh sites are deterministically or randomly distributed as assumed in these methods, thus this work discovers sites using image features with the view point of obtaining quality factors as high as possible.

Let X be a nonempty set of polygons in a Dirichlet tessellation, $x, y \in X$. A polygon $x \in X$ in a tessellation is called a **cell**. A question that arises naturally is that which sites are best or more favorable, leading to high quality cells? We will attempt to answer this in terms of the overall mesh quality factor q_{all} for the mesh cells produced by a particular set of sites. Qualities of individual mesh cells computed here are defined according to the geometry of the cell (Shewchuk, 2002; Bhatia & Lawrence, 1990a).

Let S be a set of tessellation cells, A the area of a tessellation containing a 3-sided polygon cell $s \in S$, l_1, l_2, l_3 the lengths of the sides of s with $Q(s)$ the quality of cell s . Then, for example, (Bhatia & Lawrence, 1990b), (Bank & Xu, 1996) as well as (Field, 2000) use the following smooth quality measure for a 3-sided cell.

$$Q_3(s) = 4\sqrt{3} \frac{A}{l_1^2 + l_2^2 + l_3^2}.$$

For a four sided mesh cell, the quality factor $Q_4(s)$ is defined by

$$Q_4(s) = \frac{4A}{l_1^2 + l_2^2 + l_3^2 + l_4^2}.$$

The quality $Q(s)$ of 3D tetrahedron (polyhedron with 4 sides) cell in \mathbb{R}^3 is defined by

$$Q_{43D}(s) = \frac{6\sqrt{2}V}{l_{rms}^3}.$$

Here, V is the volume of the tetrahedron and l_{rms} is given by:

$$l_{rms} = \sqrt{\frac{1}{6} \sum_{i=1}^6 l_i^2}.$$

Since the focus is on meshes of 2D surfaces in \mathbb{R}^2 , we only briefly consider 3D meshes. An overall mesh quality indicator may be defined for any meshed surface by making use of the qualities of the individual cells. One such indicator is defined by q_{all} defined below.

$$q_{all} = \frac{1}{N} \sum_{i=1}^N q_i.$$

For a plane tessellated by a set of sites S , the indicator of the overall tessellation quality is influenced by the qualities of the individual cells q_i . This provides a useful tool for discriminating sites and their tessellation quality.

Theorem 5.1. *For any plane, there exists a set of sites for which the mesh quality is maximum.*

Proof. Consider an arbitrary n-sided mesh cell. Assume the cell is a quad cell without loss of generality. For maximum q , the partial derivatives of q with respect to the l_i s should be equal.

$$\frac{\partial q}{\partial l_1} = \frac{\partial q}{\partial l_2} = \dots = \frac{\partial q}{\partial l_n}$$

$$\frac{-8Al_1}{(l_1^2 + l_2^2 + \dots + l_n^2)^2} = \frac{-8Al_2}{(l_1^2 + l_2^2 + \dots + l_n^2)^2} \dots = \frac{-8Al_n}{(l_1^2 + l_2^2 + \dots + l_n^2)^2}.$$

This happens when $l_1 = l_2 = \dots = l_n$. So generators chosen such that their half planes are equidistant from each other would satisfy this condition. □

To synthesize and crystallize the preceding deliberations on mesh quality analysis, the following algorithm is provided.

5.1. Algorithm for Mesh Quality Computation

It is clear that a Voronoï diagram is a collection of several polygons of different dimensions in accordance with the criteria already laid out, thus the quality factor of each polygon is computed as follows:

Input: set of sites S , set of cells $V(S)$

Output: Mesh Quality $Q(s)$

for each Voronoï region $V_i(s) \in V(S)$, site $s \in S$ **do**

Access the number of sides and coordinates of the vertices of the polygon;

Using the coordinates, compute the lengths l_i and Area A of the polygon;

Use l_i and A in the appropriate expression to compute cell quality $Q(s)$, $s \in S$;

end for

$Q(S) = \{Q(s)\}$ ■

The results presented in the following sections show tessellated image spaces side-by-side with the distribution of the quality factors of their cells. These are shown for image data of several subjects. From the distribution of quality factors, we obtain the mean quality measure as an indicator of overall quality of meshes due to each set of generating sites.

6. Application: Digital Image Tessellation Quality

Mesh sites are obtained using the 2D face sets from (Craw, 2009). The images are monochrome with dimensions 181 by 241 showing subjects with different expressions and orientations. The sites so obtained from them are used to tessellate the regions and subsequently, quality factors of cells are computed. The quality factors are shown in histogram plots next to the tessellated images (Fig. 2-Fig. 21).

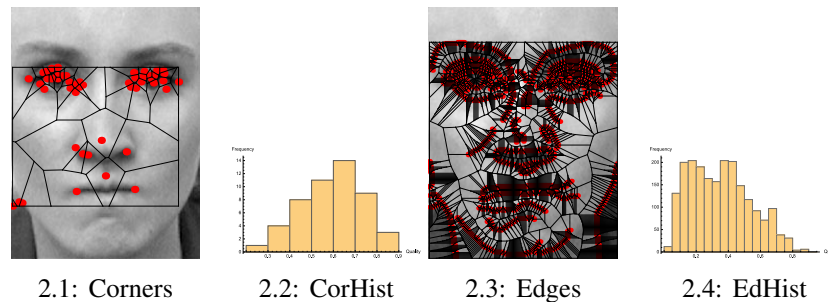


Figure 2. Corner & Edge Tessellations and Quality Histograms.

Remark 6.1. Corner-Based vs. Edge-Based Voronoi Tessellations.

A corner-based Voronoi tessellation of a face image is shown in Fig. 2.1. The corresponding corner-quality mesh histogram is given in Fig. 2.2. The horizontal axis represents mesh cell qualities and vertical axis represents the frequencies of the qualities. Because the number of image corners found are both sparse and grouped together in the facial high points representing pixel gradient changes in directions, the corresponding corner-based mesh consists of fairly large polygons surrounding the corners. Also, observe that the corner-based mesh histogram has a fairly normal distribution (skewed to the right).

An edge-based Voronoi tessellation of a face image is shown in Fig. 2.3. The corresponding edge-quality mesh histogram is given in Fig. 2.4. By contrast with image corners used as mesh generating points, the number of edge pixels found is large. In addition, the edge pixels are grouped closely together. Hence, the corresponding edge-based mesh contains many small Voronoi regions grouped closely together. The resulting edge-based mesh histogram has more than one maximum, which is an indicator that edge-based meshes have poor quality. □

Remark 6.2. Dominant-Based vs. Keypoint-Based Voronoi Tessellations.

Fig. 3.1 shows dominant-based tessellations alongside their quality measures in Fig. 3.2. Similarly, keypoint-based tessellations and their qualities are shown in Fig. 3.3 and Fig. 3.4 respectively. Dominant-based cells tend to have quality values that fall within several ranges of the quality scale. Even though dominant generators tend to have higher numbers like edge generators, they generally give higher qualities compared to edge-based cells. Keypoint-based cells like dominant-based cells tend to have their qualities falling in several ranges of quality, only that the number of fragmented ranges is usually smaller compared to that of dominant-based cells. Even

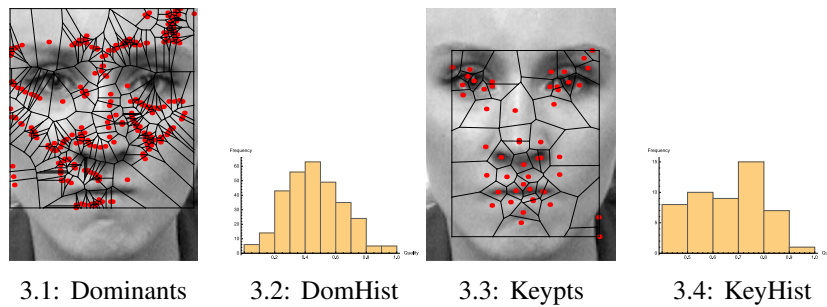


Figure 3. Dominant & Keypoint Tessellations and Quality Histograms.

though keypoint generators have smaller numbers compared to dominant generators, the location and distribution of the features favor creation of more perfect mesh cells. The qualities of dominant-based cells peak around mid scale whilst those of keypoints tend to be flat. □

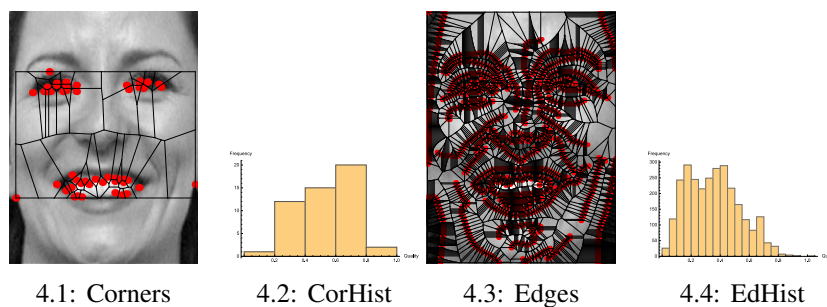


Figure 4. Corner & Edge Tessellations and Quality Histograms.

Remark 6.3. Corner-Based vs. Edge-Based Voronoi Tessellations.

The corner generators of Fig. 4.1 are concentrated in the mouth and eye regions of the subject. As a consequence, unequal distribution of the cell qualities in those regions result in Fig. 4.2. Edge-based cells on the other hand are distributed around the borders of the entire image giving peak cell quality in about mid range of the scale (see Fig. 4.3, Fig. 4.4). □

Remark 6.4. Dominant-Based vs. Keypoint-Based Voronoi Tessellations.

Dominant generators outweigh keypoints in number (see Fig. 5.1, Fig. 5.3). The generators are however concentrated around the mouth and eye regions but with extra points around the chin region in the case of dominant generators. Due to these distributions of the generators, both sets of generators have comparatively high qualities with peculiar quality factor distributions as shown in Fig. 5.2 and Fig. 5.4. □

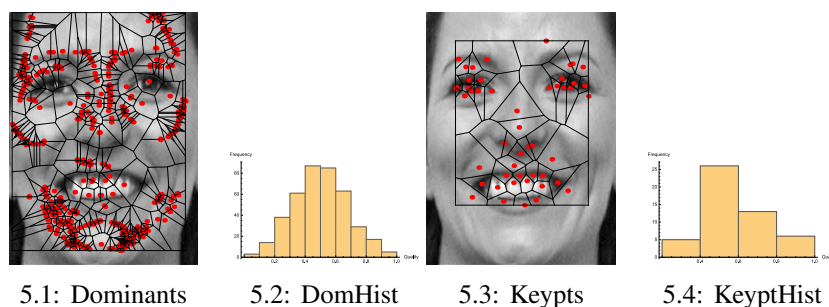


Figure 5. Dominant & Keypoint Tessellations and Quality Histograms.

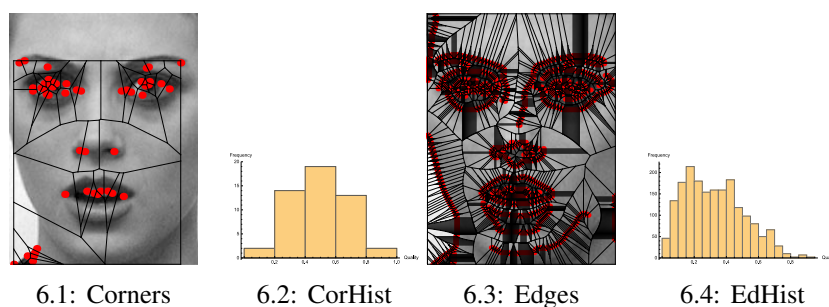


Figure 6. Corner & Edge Tessellations and Quality Histograms.

Remark 6.5. Corner-Based vs. Edge-Based Voronoi Tessellations.

Sets of generators of a subject are shown in Fig. 6.1 and Fig. 6.3. In addition to generators around the eye and mouth regions, generators around the nose and lower neck areas are returned for corner sites. Edges on the other hand are returned for regions of feature discontinuities. The quality plots of Fig. 6.2 and Fig. 6.4 represent the cells with more well laid out features on one hand and clustered generators on the other. □

Remark 6.6. Dominant-Based vs. Keypoint-Based Voronoi Tessellations.

Dominant generators returned here exclude most of the keypoint generators (Fig. 7.1, Fig. 7.3). Also, keypoint generators are more localized as opposed to the more or less global distribution of dominant generators. These fundamentally different generators produced a somewhat flat distribution of cell qualities (taken in two halves) and an alternating distribution of qualities seen in Fig. 7.4 and Fig. 7.2 respectively. □

Remark 6.7. Corner-Based vs. Edge-Based Voronoi Tessellations.

Distinct generators are returned in the case of Fig. 8.1 as opposed to less distinct ones in Fig. 8.3. Although corner generators are smaller in number, they have covered a lot of distinct and important features in the image. Edge generators however are localized to boundary regions. Given the layout of generators, the corner generators favored creating meshes approaching perfect lengths than edge generators as seen in Fig. 8.2 and Fig. 8.4 respectively. □

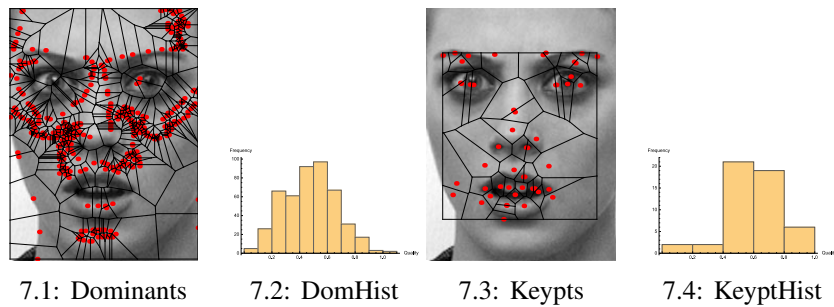


Figure 7. Dominant & Keypoint Tessellations and Quality Histograms.

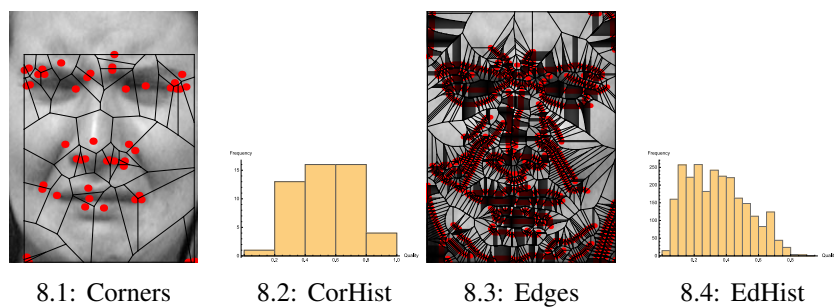


Figure 8. Corner & Edge Tessellations and Quality Histograms.

Remark 6.8. Dominant-Based vs. Keypoint-Based Voronoi Tessellations.

In Fig. 9.1 the layout of the generators does not favour polygons with equal lengths. This is evident in the fragmented nature of the quality factors in Fig. 9.2. Generators however in Fig. 9.3 performed better in their quality distributions in Fig. 9.4. □

Remark 6.9. Corner-Based vs. Edge-Based Voronoi Tessellations.

Examine for a brief moment the corner and edge generators in Fig. 10.1 and Fig. 10.3 alongside their quality factors in Fig. 10.2 and Fig. 10.4 respectively. Notice that some generators are clustered around the eye regions. The situation however is still better in terms of affording better overall quality as opposed to similar clustering of edge detectors in several areas. □

Remark 6.10. Dominant-Based vs. Keypoint-Based Voronoi Tessellations.

Dominants and their tessellations in Fig. 11.1 occupy a larger area compared to keypoints and their tessellations in Fig. 11.3. Although many cells result in Fig. 11.2, most of the quality factors are concentrated in the first half of the scale. Keypoints on the other hand are better laid out and although of a smaller number, they cover a comparable space and give a higher overall quality measure from Fig. 11.4. □

A complete set of results based on centroids of Voronoi regions specified by corner, edge, dominant and keypoint tessellations is shown in Fig. 12-Fig. 21. For those results shown, generators have been obtained corresponding to Voronoi regions of corner, edge, dominant and keypoint

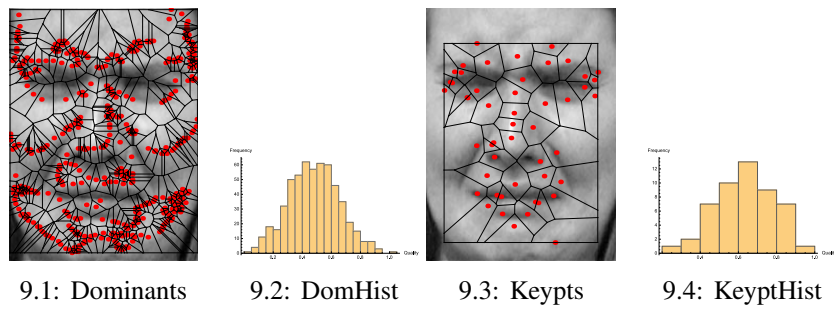


Figure 9. Dominant & Keypoint Tessellations and Quality Histograms.

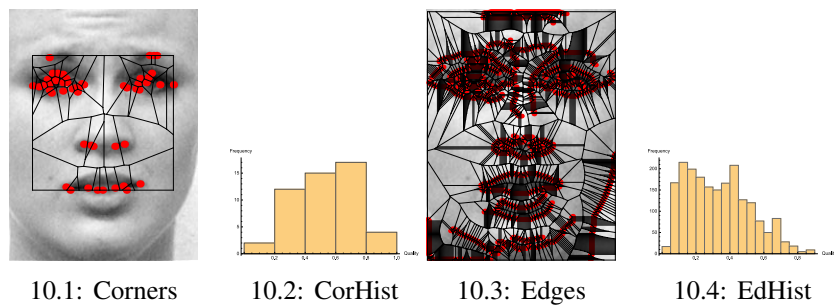


Figure 10. Corner & Edge Tessellations and Quality Histograms.

tessellations. These are the centroids of Voronoi regions in Fig. 2-Fig. 11. In the following text, remarks are included pertaining to the regions and their qualities.

Remark 6.11. Corner centroid-Based vs. Edge centroid-Based Voronoi Tessellations.

In comparing Fig. 2.1 to Fig. 12.1 we notice that the numbers of generators is the same. However, an interesting situation arises. The polygons in the latter case have a reduced variability in their lengths. This led to better quality measures with quality distributions as in Fig. 12.2. In a similar vein the number of generators are the same for edge generators and edge-centroid generators. The overall quality has been improved from the neighborhoods of 0.3 to above 0.5 (Fig. 12.3, Fig. 12.4).

□

Remark 6.12. Dominant centroid-Based vs. Keypoint-Based Voronoi tessellations.

Notice that the locations of centroid generators are different from those of dominant generators. This distribution led to a mesh covering of about three quarters of the image as seen in Fig. 13.1. These generators favored better mesh qualities with the distribution seen in Fig. 13.2. Keypoint centroids of Fig. 13.3 and their tessellations however cover comparable areas with keypoint generators. The use of the centroid generators improved the mesh qualities as shown in Fig. 13.4.

□

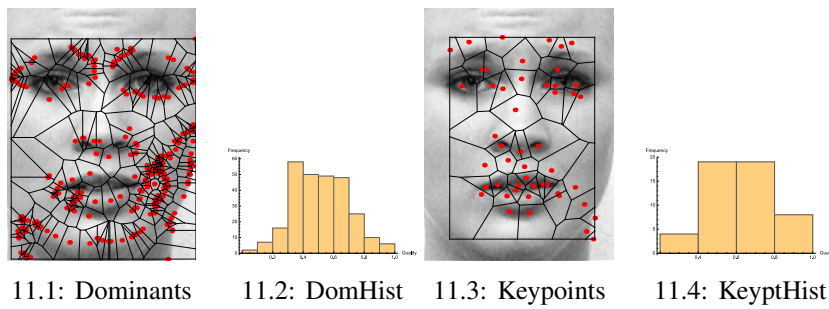


Figure 11. Dominant & Keypoint Tessellations and Quality Histograms.

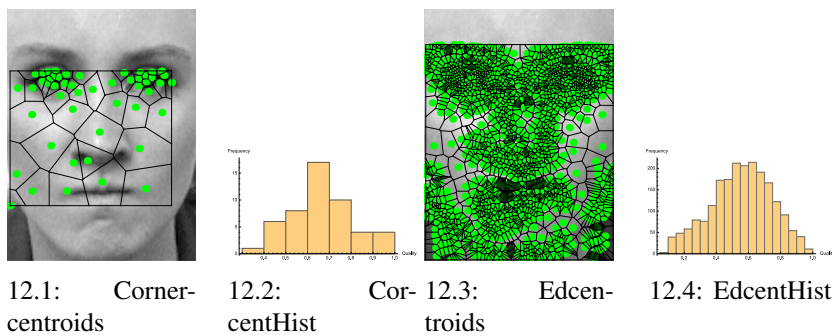


Figure 12. Corner & Edge Centroid Tessellations and Quality Histograms.

Remark 6.13. Corner centroid-Based vs.Edge centroid-Based Voronoi Tessellations.

The centroid sites of Fig. 14.1 barely covered the eyes, nose and most of the mouth region. As expected, edge centroids cover and tessellate the entire image Fig. 14.3. Although the mesh qualities are improved, they are consistent with distribution of cell lengths in Fig. 14.2 and Fig. 14.3. □

Remark 6.14. Dominant centroid-Based vs.Keypoint-Based Voronoi tessellations.

Observe in the tessellated spaces that the dominant centroid generators are mostly clustered in all areas except in the eyes and the mouth (see Fig. 15.1). Keypoint centroids however are distributed primarily around the facial features such as the mouth, nose and mouth as observed in Fig. 15.3. These generators are less clustered compared to their counterparts for dominant and keypoint generator tessellations. With the favorable condition for improved cell qualities obtained by using the centroids, the qualities are distributed across the entire quality scale as seen in Fig. 15.2 and Fig. 15.4. In most of the cases, the minimum cell quality for keypoint centroid-based generators is in the neighborhoods of 0.4-0.5. □

Remark 6.15. Corner centroid-Based vs.Edge centroid-Based Voronoi Tessellations.

Observe that several generators are located in the eye ball regions in Fig. 16.1. The tessellated regions cover up to the chin region. Less varying polygonal lengths led to distribution of mesh

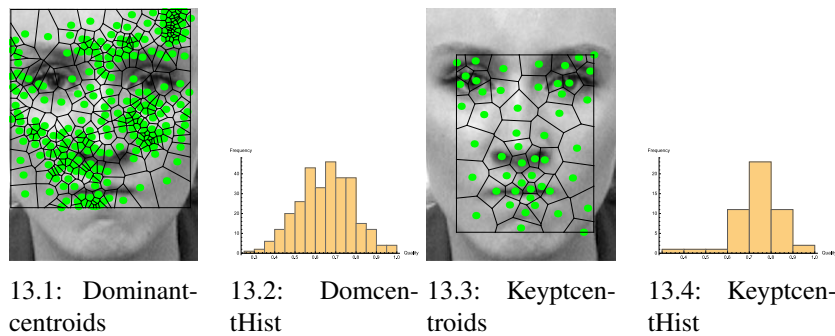


Figure 13. Dominant & Keypoint Centroid Tessellations and Quality Histograms.

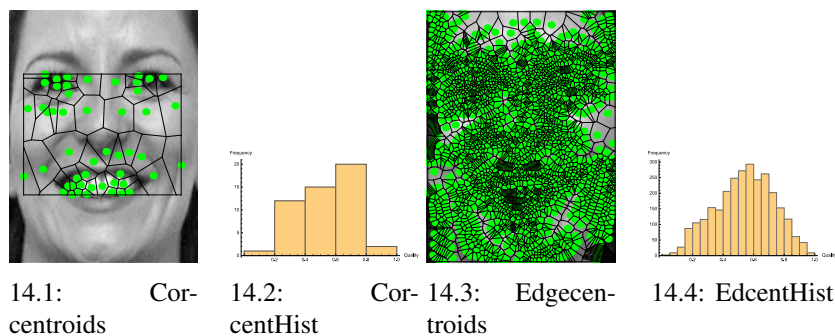


Figure 14. Corner & Edge Centroid Tessellations and Quality Histograms.

qualities in Fig. 16.2. Most of the image plane is however covered by edge centroid generators (Fig. 16.3). Although the quality factors are fragmented, they cover the entire scale with the distribution shown in Fig. 16.4 with minimum quality starting at about 0.3. □

Remark 6.16. Dominant centroid-Based vs.Keypoint-Based Voronoï tessellations.

Dominant generators cover most of the image plane. However, most of the generators tend to be concentrated just below the eyes and nose regions. Also notice that regions without clustering of the cells tend to be polygons whose lengths tend to be equal. This distribution of the generators affords cells and qualities in Fig. 17.1 and Fig. 17.2. Although there are keypoint centroid generators in the eye, nose and mouth regions, they are not clustered as in the previous case (see Fig. 17.3). They are better spaced out giving the qualities in Fig. 17.4. □

Remark 6.17. Corner centroid-Based vs.Edge centroid-Based Voronoï Tessellations.

Centroid corner generators tessellate the image region around the facial features. This placement of the generators yielded cells with flat distribution of qualities across the scale (see Fig. 18.1, Fig. 18.2). Edge centroid generators on the other hand tessellated the entire image but with the sites clustered together in most areas. Even though the whole image space is covered, the resulting cells do not promote better overall quality as compared with centroidal corner tessellations (see Fig. 18.3, Fig. 18.4). □

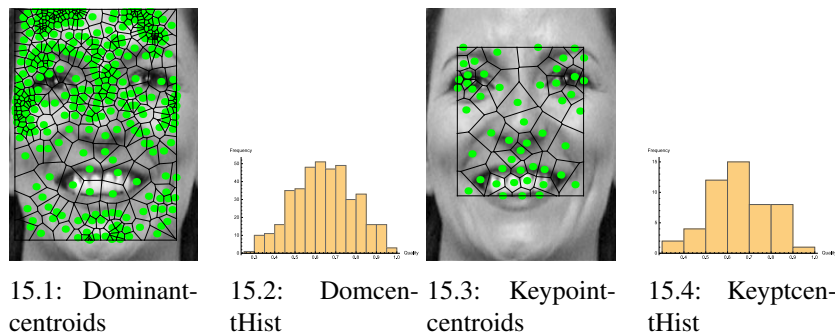


Figure 15. Dominant & Keypoint Centroid Tessellations and Quality Histograms.

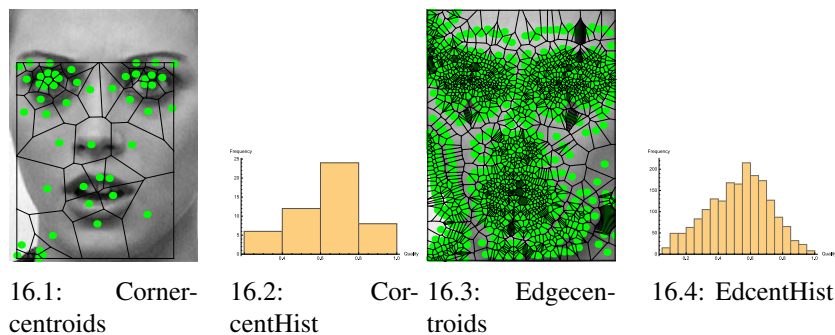


Figure 16. Corner & Edge Centroid Tessellations and Quality Histograms.

Remark 6.18. Dominant centroid-Based vs.Keypoint-Based Voronoï tessellations.

Dominant generators tessellated most of the image with cells of improved qualities compared to previous situations (Fig. 19.1 and Fig. 19.2). The qualities of the cells cover the entire scale in both scenarios. However, you would notice that the cells of Fig. 19.3 are of better quality Fig. 19.4. □

Remark 6.19. Corner centroid-Based vs.Edge centroid-Based Voronoï Tessellations.

Cell qualities cover the entire scale in Fig. 20.2 and Fig. 20.4. The difference however lies in the numbers of the generators and their positions as seen in Fig. 20.1 and Fig. 20.3. □

Remark 6.20. Dominant centroid-Based vs.Keypoint-Based Voronoï tessellations.

Although common generators are returned in Fig. 21.1 and Fig. 21.3, the concentration of points in the left cheek region and the lower neck region of the test subject favored better mesh qualities generation as seen in comparing Fig. 21.2 and Fig. 21.4. □

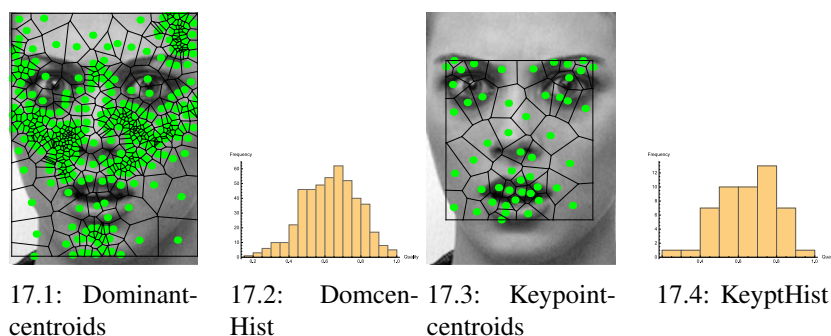


Figure 17. Dominant & Keypoint Centroid Tessellations and Quality Histograms.

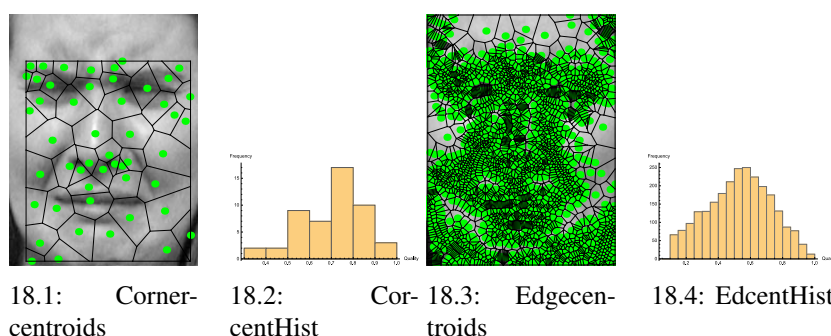


Figure 18. Corner & Edge Centroid Tessellations and Quality Histograms.

The overall quality measures of meshes for each test subject class based on corner, edge, dominant and keypoint sites is presented in Fig. 22. For several images of the same subject, q_{all} is computed for each set of generators. The plot therefore presented shows the relationships of sets of generators and the overall quality of meshes for tessellated images. In the plot, the trend indicates that for sets of generators and their tessellations, keypoints give the meshes with the highest qualities. This is due to the distribution of the sites in such a way that they tend to produce perfect polygons. Edge generators on the other hand consistently give low quality tessellations. This is the case because edge sites tend to be clustered together thus producing qualities on the lower side of the scale. The qualities of corner and dominant generators assume a place in between those of edge and keypoint tessellations. The qualities of the cells by sets of generators is in proportion to distributions that tend to give perfect polygons.

The method of centroids of regions defined by image centroids shows how mesh qualities may be improved (Fig. 23). In this figure, the qualities of the cells have been improved for all sets of generators. However, the order of mesh qualities has been preserved. This improvement results from the energy minimization property of centroids and the quasi perfect polygons centroids tend to produce.

Remark 6.21. What High Quality Meshes Reveal About Tessellated Images.

Sets of sites are used to generate a Voronoi diagram (also called a mesh) on a digital image. Each

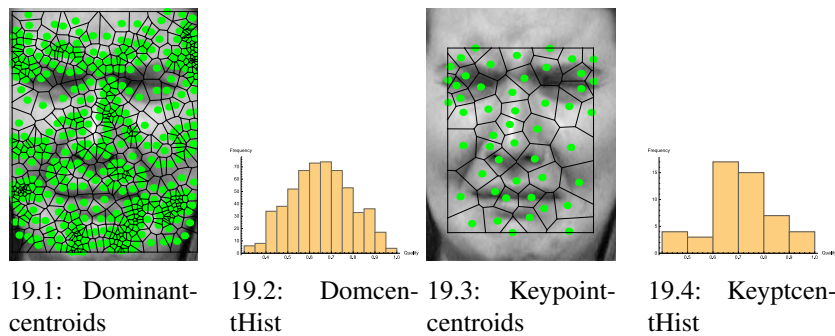


Figure 19. Dominant & Keypoint Centroid Tessellations and Quality Histograms.

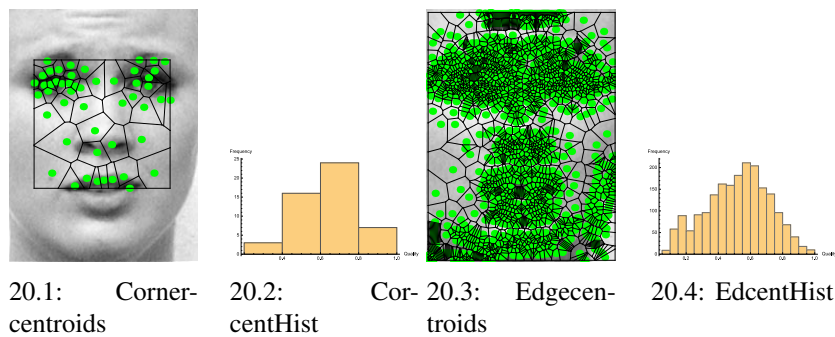


Figure 20. Corner & Edge Centroid Tessellations and Quality Histograms.

region of a site is convex set represented geometrically by a polygon.

Associated with a set of sites are the qualities of the individual cells and their overall quality measure. So given sets of generators and overall tessellation qualities, the tessellation quality characterizes the underlying local structure of a collection of Voronoï regions. Quality of Voronoï polygons give us shape information about the region they cover. For the tessellated spaces, notice that the numbers of interior Voronoï polygons is greater than open border polygons. This shows that the mesh generation patterns are globular in nature. For example, the following quality expressions yielding $q = 1$ would indicate the presence of equilateral triangle and perfect quadrilateral respectively.

$$q = \frac{4\sqrt{3}A}{l_1^2 + l_2^2 + l_3^2}$$

$$q = \frac{4A}{l_1^2 + l_2^2 + l_3^2 + l_4^2}$$

For small quality measurements as seen for edge point patterns sets, the measures are an indicator that the generators are on a curve and are closely spaced.

The qualities of cells and the overall quality of a tessellation characterizes the regularity and repeatability of a mesh generator set. If the space is covered with individual cells all of unit

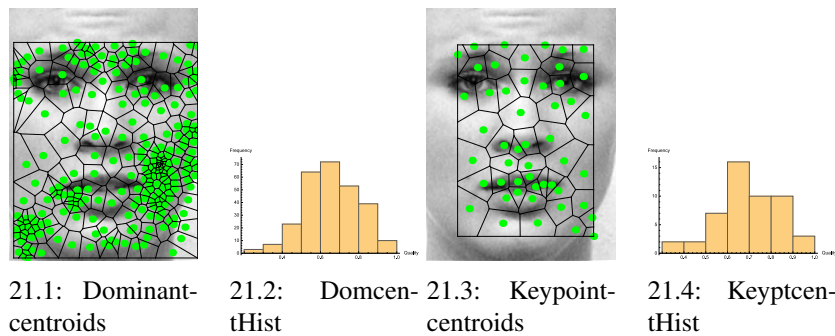


Figure 21. Dominant & Keypoint Centroid Tessellations and Quality Histograms.

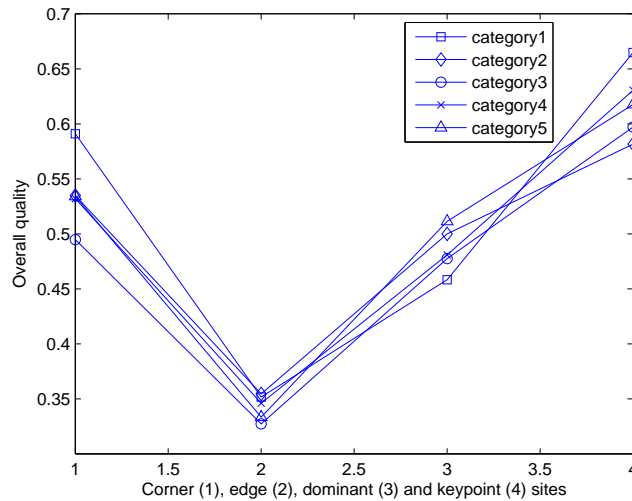


Figure 22. (a) Plot of overall quality factors against choice of sites for image categories.

quality, it indicates that the pattern points produce perfect polygons in the space. Associated with this is the simplicity of the design of the underlying pattern. Higher qualities indicate simple and predictable distributions while the converse holds for low qualities. This reveals the regularity of the points in the distribution of the pattern set. It also follows that the density of the points is uniform in the plane of the pattern space.

The quality of mesh cells and their associated image spaces give information on the separability of dot patterns. Generally, the higher the quality the greater the separation between points in the set. For example, the quality of edge generators is small compared to those of corners, keypoints and dominant generators and hence the separation of edge point pattern sets is poor compared to corner, keypoint and dominant pattern sets.

An extension of the separability of dot patterns and their associated factors is the notion of how adequately the point pattern represents the image space. For example if keypoints are used as mesh generators, then the generated mesh contains a distribution of polygons that surround objects in

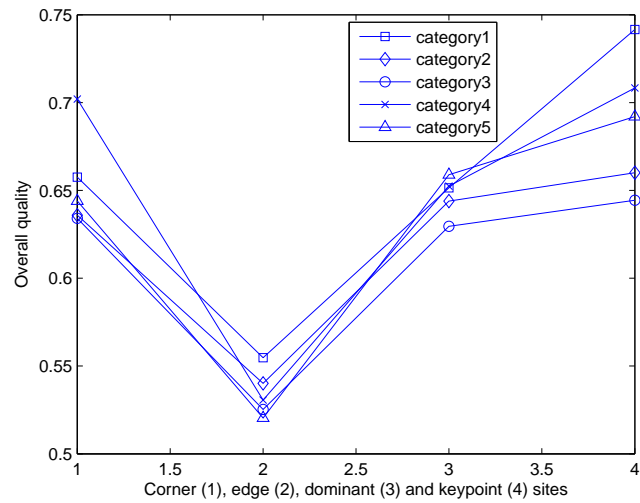
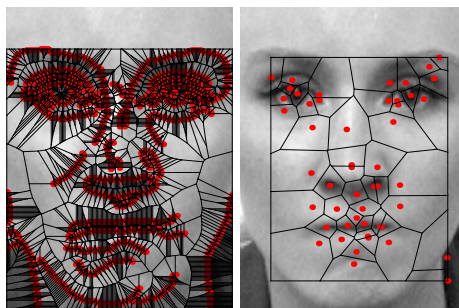


Figure 23. (a) Plot of overall quality factors against centroids of sites for image categories.

an image. In other words, the high quality of a keypoint-based mesh yields more information about image objects. To illustrate, edge tessellations give meshes with overall quality of 0.352 versus 0.665 for keypoint tessellations in Fig. 24.1 and Fig. 24.2 respectively.

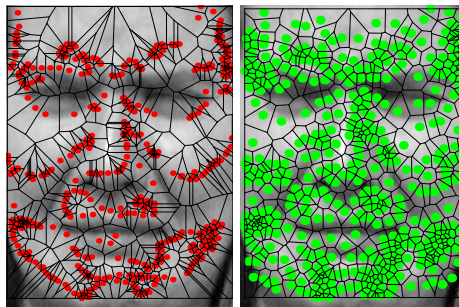


24.1: Sufficient1 24.2: Sufficient2

Figure 24. Meshes demonstrating sufficiency of coverings

Another important piece of information furnished by meshes relates to symmetry. The higher the quality, the greater the symmetry of image objects. High quality meshes tend to have connected sets of highly symmetrical polygons. To demonstrate symmetry, consider two generators $S_1(x_1, y_1)$ and $S_2(x_2, y_2)$ on either side of a vertical line, y through a nose point. The generators S_1 and S_2 are symmetrical if and only if $\|(x_1, y_1), y\| = \|(x_2, y_2), y\|$. The mesh coverings in Fig. 25 are due to dominant generators and their centroids. If you draw a vertical line through the center of the nose region, you would notice that the generators in Fig. 25.2 demonstrate a better reflection of features than in Fig. 25.1. Symmetry thus furnishes us with a tool for feature location given features on one

half of the space.



25.1: Symmetry1 25.2: symmetry2

Figure 25. Symmetry of features

Last but not least, the quality of an image mesh covering may be used to estimate image quality. Two image quality assessment methods will be compared here: Image structural similarity index (SSIM) and image quality through Voronoi tessellations. SSIM compares normalized local pixel patterns (Wang et al., 2004). For a signal pair x, y it is defined by (Wang et al., 2004)

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)}$$

In the definition above, the SSIM between x and y uses signal statistics; the mean values of the signals μ_x, μ_y , their variances σ_x^2, σ_y^2 , cross correlation between signals σ_{xy} and constants C_1 and C_2 .

Voronoi mesh image quality on the other hand is defined by using the geometry of the polygons enclosing image object points and regions in a tessellated space. Given the q measures of a tessellated image space, the image quality is defined using q_{all} .

Notice that $SSIM(x, y)$ uses the entire image spaces for image quality assessment and the images must be of the same size. Besides the size constraint, huge signal sizes can make it a computationally intensive approach. Voronoi analysis of image quality on the other hand uses a small set of the features used in SSIM. To demonstrate, four image signals and their mesh coverings are given in Fig. 26 and Fig. 27 respectively.

Table 1. SSIM and Quality Indexes

Image	SSIM	q_{all}
1	0.5569	0.562207
2	-	0.581664
3	0.5969	0.538463
4	-	0.538030

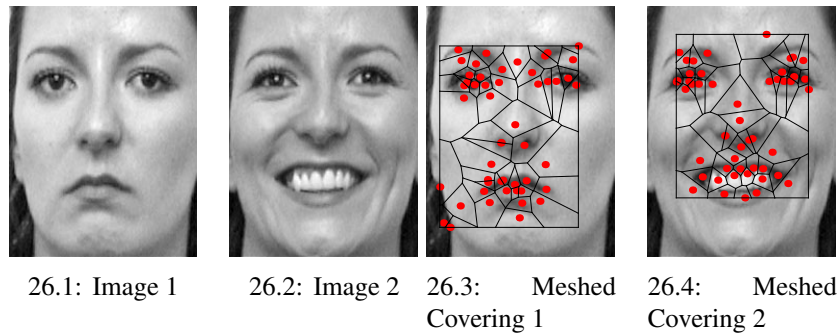


Figure 26. An image pair and their meshed domains for SSIM and quality comparisons

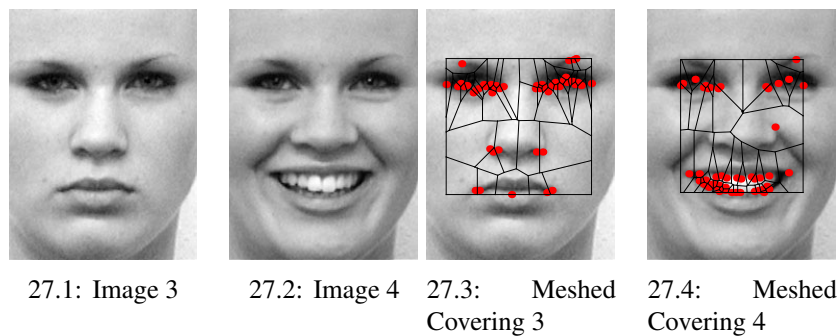


Figure 27. An image pair and their meshed domains for SSIM and quality comparisons

Image quality indicators of the signals in Fig. 26 and Fig. 27 is reported in Table. 1. Note that although the entire feature space is utilized in the calculation of the SSIM, they are comparable with those obtained with Voronoï tessellations. Notwithstanding, the Voronoï image quality method provides a quality indicator for each image whereas two signals are needed to output their SSIM. The ideal index possible is 1. An image Voronoï index of 1 means that the point pattern spatial geometry or arrangement is perfectly regular while an SSIM of 1 would mean a perfect match between the two images.

□

Note that there are differences between the SSIM and q_{all} indexes, although small. The differences stem from the fact that the mesh approach uses polygonal regions and their shape information to capture image quality, whilst the SSIM index depends on all pixel values in the image. Also, the SSIM index depends on constants, as it affects the indicators. The mesh approach however depends on image features only.

Note that mesh qualities show that low overall quality generators are not sufficient descriptors of image features. On the other hand high mesh qualities indicate important and influential image

features useful for image and mesh analysis. Also, note that the numbers of generator sites are different for corner, edge, dominant and keypoint sites. Even though edges tend to have the highest numbers, their qualities are very poor in comparison to the other generators. This shows that high quality meshes such as those generated by keypoints identify better the most influential, more well laid out and important features in image characterization and representation.

7. Conclusion

Voronoi generating points useful in image tessellations and visual image quality analysis have been identified. Previous works focused on generating points based on random distributions, sites without consideration of feature locations with scant attention given to resulting mesh quality. Various mathematical results pertaining to meshes, quality have been identified and proved. Centroidal tessellations have been used as a means to improve tessellation quality. This appears to be a new way of tessellation quality improvement as the literature hardly considers feature-based centroidal tessellations and resulting qualities. The measurement of image mesh quality offers a means of choosing suitable mesh generating points or sites based on their sufficiency in characterizing features of digital images. An important limitation of the model here is that slight perturbation of generators would mean changing locations leading to possibly different polygonal lengths and areas, hence the need to readjust or compute quality measures. Choosing a subset of the entire signal space for generators does not seem to be a significant limitation since it usually covers a significant portion of the space. However, we can always increase the numbers of generators to cover larger spaces although at higher computational costs.

References

- Agrell, Erik (1996). Voronoi regions for binary linear block codes. *Information Theory, IEEE Transactions on* **42**(1), 310–316.
- Ahuja, Narendra and Bruce Jay Schachter (1982). *Pattern models*. John Wiley & Sons Inc.
- Arbeláez, Pablo A and Laurent D Cohen (2006). A metric approach to vector-valued image segmentation. *International Journal of Computer Vision* **69**(1), 119–126.
- Aurenhammer, Franz (1991). Voronoi diagrams - a survey of a fundamental geometric data structure. *ACM Computing Surveys (CSUR)* **23**(3), 345–405.
- Bank, R.E. and J. Xu (1996). An algorithm for coarsening unstructured meshes. *Numerische Mathematik* **73**, 1–36.
- Bern, Marshall and P Plassmann (1999). Mesh generation. *Handbook of Computational Geometry*.
- Bhatia, RP and KL Lawrence (1990a). Two-dimensional finite element mesh generation based on stripwise automatic triangulation. *Computers & Structures* **36**(2), 309–319.
- Bhatia, R.P. and K.L. Lawrence (1990b). Two-dimensional finite element mesh generation based on stripwise automatic triangulation. *Computers & Structures* **36**(2), 309–319.
- Brauwerman, Roger, Sarah Joy Zoll, Christopher L Farmer and Max Gunzburger (1999). Centroidal voronoi tessellations are not good jigsaw puzzles. <http://www.math.iastate.edu/reu/1999/cvt.pdf>.
- Brown, Matthew and David G Lowe (2002). Invariant features from interest point groups.. In: *BMVC*. number s 1.
- Burns, Jared (2009). Centroidal voronoi tessellations. <https://www.whitman.edu/Documents/Academics/Mathematics/burns.pdf>.
- Canny, John (1986). A computational approach to edge detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on* (6), 679–698.

- Craw, Ian (2009). 2d face sets. http://pics.stir.ac.uk/2D_face_sets.htm. Pain Expression Subset.
- Demir, Cigdem and Bülent Yener (2005). Automated cancer diagnosis based on histopathological images: a systematic survey. *Rensselaer Polytechnic Institute, Tech. Rep.*
- Du, Qiang, Vance Faber and Max Gunzburger (1999). Centroidal voronoi tessellations: applications and algorithms. *SIAM review* **41**(4), 637–676.
- Ebeida, Mohamed S and Scott A Mitchell (2012). Uniform random voronoi meshes. In: *Proceedings of the 20th International Meshing Roundtable*. pp. 273–290. Springer.
- Ebeida, Mohamed S, Scott A Mitchell, Andrew A Davidson, Anjul Patney, Patrick M Knupp and John D Owens (2011). Efficient and good delaunay meshes from random points. *Computer-Aided Design* **43**(11), 1506–1515.
- Feng, Xiaoyi, Yangming Lai, Xiaofei Mao, Jinye Peng, Xiaoyue Jiang and Abdenour Hadid (2013). Extracting local binary patterns from image key points: Application to automatic facial expression recognition. In: *Image Analysis*. pp. 339–348. Springer.
- Field, D.A. (2000). Qualitative measures for initial meshes. *Int. J. for Numerical Methods in Engineering* **47**, 887–906.
- Förstner, Wolfgang and Eberhard Gülch (1987). A fast operator for detection and precise location of distinct points, corners and centres of circular features. In: *Proc. ISPRS intercommission conference on fast processing of photogrammetric data*. pp. 281–305.
- George, Paul Louis (2006). Adaptive mesh generation in 3 dimensions by means of a delaunay based method. applications to mechanical problems. In: *III European Conference on Computational Mechanics*. Springer. pp. 18–18.
- Guru, DS and R Dinesh (2004). Non-parametric adaptive region of support useful for corner detection: a novel approach. *Pattern Recognition* **37**(1), 165–168.
- Hamanaka, Masatoshi and Keiji Hirata (2002). Applying voronoi diagrams in the automatic grouping of polyphony. *Information Technology Letters* **1**(1), 101–102.
- Harris, Chris and Mike Stephens (1988). A combined corner and edge detector.. In: *Alvey vision conference*. Vol. 15. Manchester, UK. p. 50.
- Kang, Sung Kwan, Young Chul Choung and Jong An Park (2005). Image corner detection using hough transform. In: *Pattern Recognition and Image Analysis*. pp. 279–286. Springer.
- Kim, Bongjoe, Jihoon Choi, Yongwoon Park and Kwanghoon Sohn (2012). Robust corner detection based on image structure. *Circuits, Systems, and Signal Processing* **31**(4), 1443–1457.
- Kitchen, Les and Azriel Rosenfeld (1980). Gray-level corner detection. Technical report. DTIC Document.
- Knupp, Patrick M (2001). Algebraic mesh quality metrics. *SIAM journal on scientific computing* **23**(1), 193–218.
- Koenderink, Jan J and Whitman Richards (1988). Two-dimensional curvature operators. *JOSA A* **5**(7), 1136–1141.
- Leibon, Greg and David Letscher (2000). Delaunay triangulations and voronoi diagrams for riemannian manifolds. In: *Proceedings of the sixteenth annual symposium on Computational geometry*. ACM. pp. 341–349.
- Lindeberg, Tony (1993). *Scale-space theory in computer vision*. Springer Science & Business Media.
- Lindeberg, Tony (1998). Feature detection with automatic scale selection. *International journal of computer vision* **30**(2), 79–116.
- Lindeberg, Tony (2008). *Scale-Space*. Wiley Online Library.
- Liu, Jinyi and Shuang Liu (2004). A survey on applications of voronoi diagrams. *Journal of Engineering Graphics* **22**(2), 125–132.
- Lowe, David G (1999). Object recognition from local scale-invariant features. In: *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*. Vol. 2. Ieee. pp. 1150–1157.
- Lowe, David G (2004). Distinctive image features from scale-invariant keypoints. *International journal of computer vision* **60**(2), 91–110.
- McLean, Alex (2007). Voronoi diagrams of music. URL <http://doc.gold.ac.uk/~ma503am/essays/voronoi/voronoi-diagrams-of-music.pdf>. Accessed.

- Mikolajczyk, Krystian and Cordelia Schmid (2004). Scale & affine invariant interest point detectors. *International journal of computer vision* **60**(1), 63–86.
- Mitchell, HB (2010). Image key points. In: *Image Fusion*. pp. 163–166. Springer.
- Mitchell, Scott A (1993). Mesh generation with provable quality bounds. Technical report. Cornell University.
- Møller, Jesper and Øivind Skare (2001). Coloured voronoi tessellations for bayesian image analysis and reservoir modelling. *Statistical modelling* **1**(3), 213–232.
- Moravec, Hans P (1980). Obstacle avoidance and navigation in the real world by a seeing robot rover.. Technical report. DTIC Document.
- Owen, Steven J (1998). A survey of unstructured mesh generation technology. In: *IMR*. pp. 239–267.
- Pan, Haixia, Yanxiang Zhang, Chunlong Li and Huafeng Wang (2014). An adaptive harris corner detection algorithm for image mosaic. In: *Pattern Recognition*. pp. 53–62. Springer.
- Park, Seung Jin, Muhammad Bilal Ahmad, Rhee Seung-Hak, Seung Jo Han and Jong An Park (2004). Image corner detection using radon transform. In: *Computational Science and Its Applications–ICCSA 2004*. pp. 948–955. Springer.
- Peraire, Jaime, Morgan Vahdati, Ken Morgan and Olgierd C Zienkiewicz (1987). Adaptive remeshing for compressible flow computations. *Journal of computational physics* **72**(2), 449–466.
- Persson, Per-Olof (2004). Mesh generation for implicit geometries. PhD thesis. Citeseer.
- Persson, Per-Olof and Gilbert Strang (2004). A simple mesh generator in matlab. *SIAM review* **46**(2), 329–345.
- Peters, J.F. (2015a). Proximal Delaunay triangulation regions. *PJMS [Proc. Jangjeon Math. Soc.]* pp. 1–10. *accepted*.
- Peters, J.F. (2015b). Proximal Voronoi regions, convex polygons, & Leader uniform topology. *Advances in Math.: Sci. J.* **4**(1), 1–5.
- Peters, J.F. (2015c). Visibility in proximal Delaunay meshes and strongly near Wallman proximity. *Advances in Math.: Sci. J.* **4**(1), 41–47.
- Rajan, V.T. (1994). Optimality of the delaunay triangulation in \mathbb{R}^d . *Discrete & Computational Geometry* **12**(1), 189 – 202.
- Ramella, Massimo, Mario Nonino, Walter Boschin and Dario Fadda (1998). Cluster identification via voronoi tessellation. *arXiv preprint astro-ph/9810124*.
- Rivara, Maria-Cecilia (1984). Mesh refinement processes based on the generalized bisection of simplices. *SIAM Journal on Numerical Analysis* **21**(3), 604–613.
- Rosten, Edward and Tom Drummond (2006). Machine learning for high-speed corner detection. In: *Computer Vision–ECCV 2006*. pp. 430–443. Springer.
- Ruppert, Jim (1995). A delaunay refinement algorithm for quality 2-dimensional mesh generation. *Journal of algorithms* **18**(3), 548–585.
- Shewchuk, J (2002). What is a good linear finite element? interpolation, conditioning, anisotropy, and quality measures (preprint). *University of California at Berkeley*.
- Smith, Stephen M. and J. Michael Brady (1997). SUSAN - a new approach to low level image processing. *International journal of computer vision* **23**(1), 45–78.
- Voronoi, G. (1903). Sur un problème du calcul des fonctions asymptotiques. *J. für die reine und angewandte Math.* **126**, 241–282. JFM 38.0261.01.
- Voronoi, G. (1907). Nouvelles applications des paramètres continus à la théorie des formes quadratiques. premier mémoire. *J. für die reine und angewandte Math.* **133**, 97–178.
- Voronoi, G. (1908). Sur un problème du calcul des fonctions asymptotiques. *J. für die reine und angewandte Math.* **134**, 198–287. JFM 39.0274.01.
- Wang, Han and Michael Brady (1995). Real-time corner detection algorithm for motion estimation. *Image and Vision Computing* **13**(9), 695–703.

Wang, Zhou, Alan C Bovik, Hamid R Sheikh and Eero P Simoncelli (2004). Image quality assessment: From error visibility to structural similarity. *Image Processing, IEEE Transactions on* **13**(4), 600–612.

Woźniak, Marcin and Zbigniew Marszałek (2014). An idea to apply firefly algorithm in 2d image key-points search. In: *Information and Software Technologies*. pp. 312–323. Springer.



Initial Maclaurin Coefficients Bounds for New Subclasses of Bi-univalent Functions

Basem Aref Frasin^{a,*}, Tariq Al-Hawary^b

^a*Department of Mathematics, Al al-Bayt University, Mafraq, Jordan*
^b*Department of Mathematics, Al al-Bayt University, Mafraq, Jordan.*

^b*Department of Applied Science, Ajloun College, Al-Balqa Applied University, Ajloun 26816, Jordan.*

Abstract

In this work we introduce the subclasses $\mathcal{L}_\Sigma(\theta, \alpha)$ and $\mathcal{L}_\Sigma(\theta, \gamma)$ of bi-univalent functions. Furthermore, we obtain coefficient bounds involving the Taylor-Maclaurin coefficients $|a_2|$ and $|a_3|$ for functions belonging to these classes. The results presented in this paper would generalize those in related works of several earlier authors.

Keywords: Analytic and univalent functions, Bi-univalent functions, Starlike and convex functions, Coefficients bounds.

2010 MSC: 30C45, 30C50.

1. Introduction and preliminaries

Let \mathcal{A} be the class of functions f which are analytic in the open unit disk $\mathcal{U} = \{z : |z| < 1\}$ with the conditions $f(0) = 0$ and $f'(0) = 1$ and having form

$$f(z) = z + a_2z^2 + a_3z^3 + \dots \quad (z \in \mathcal{U}). \quad (1.1)$$

Further, by \mathcal{S} we will denote the class of all functions in \mathcal{A} which are univalent in \mathcal{U} .

For each θ , $-\pi < \theta \leq \pi$, Silverman and Silvia (Silverman & Silvia, 1999) introduced the class

$$\mathcal{L}(\theta) = \left\{ f \in \mathcal{A} : \operatorname{Re} \left(f'(z) + \frac{1 + e^{i\theta}}{2} z f''(z) \right) > 0, \quad z \in \mathcal{U} \right\}$$

and they proved that $\mathcal{L}(\theta) \subset \mathcal{L}(\pi)$, where $\mathcal{L}(\pi)$ is the well known class \mathcal{R} that consists of univalent functions in whose derivatives have positive real part in \mathcal{U} (Alexander, 1915). The class $\mathcal{L}(0)$

*Corresponding author

Email addresses: bafrasin@yahoo.com (Basem Aref Frasin), tariq_amh@yahoo.com (Tariq Al-Hawary)

was studied by Singh and Singh (Singh & Singh, 1989), Lewandowski et al. (Lewandowski et al., 1976), Chichra (Chichra, 1977), and Silverman (Silverman, 1994).

It is well known that every function $f \in \mathcal{S}$ has an inverse f^{-1} , defined by

$$f^{-1}(f(z)) = z \quad (z \in \mathcal{U})$$

and

$$f(f^{-1}(w)) = w \quad (|w| < r_0(f); r_0(f) \geq \frac{1}{4})$$

where

$$f^{-1}(w) = w - a_2w^2 + (2a_2^2 - a_3)w^3 - (5a_2^3 - 5a_2a_3 + a_4)w^4 + \dots .$$

A function $f \in \mathcal{A}$ is said to be bi-univalent in \mathcal{U} if both $f(z)$ and $f^{-1}(z)$ are univalent in \mathcal{U} .

Let Σ denote the class of bi-univalent functions in \mathcal{U} given by (1.1). For a brief history and interesting examples in the class Σ , see (Srivastava et al., 2010).

Brannan and Taha (Brannan & Taha, 1988) (see also (Taha, 1981)) introduced certain subclasses of the bi-univalent function class Σ similar to the familiar subclasses $\mathcal{S}^*(\alpha)$ and $\mathcal{K}(\alpha)$ of starlike and convex functions of order $\alpha(0 \leq \alpha < 1)$, respectively (see (Brannan & Taha, 1988)). Thus, following Brannan and Taha (Brannan & Taha, 1988) (see also (Taha, 1981)), a function $f \in \mathcal{A}$ is in the class $\mathcal{S}_{\Sigma}^*[\alpha]$ of strongly bi-starlike functions of order $\alpha(0 < \alpha \leq 1)$ if each of the following conditions is satisfied:

$$f \in \Sigma \text{ and } \left| \arg \left(\frac{zf'(z)}{f(z)} \right) \right| < \frac{\alpha\pi}{2} \quad (0 < \alpha \leq 1, z \in \mathcal{U})$$

and

$$\left| \arg \left(\frac{wg'(w)}{g(w)} \right) \right| < \frac{\alpha\pi}{2} \quad (0 < \alpha \leq 1, w \in \mathcal{U}),$$

where g is the extension of f^{-1} to \mathcal{U} . The classes $\mathcal{S}_{\Sigma}^*(\alpha)$ and $\mathcal{K}_{\Sigma}(\alpha)$ of bi-starlike functions of order α and bi-convex functions of order α , corresponding (respectively) to the function classes $\mathcal{S}^*(\alpha)$ and $\mathcal{K}(\alpha)$, were also introduced analogously. For each of the function classes $\mathcal{S}_{\Sigma}^*(\alpha)$ and $\mathcal{K}_{\Sigma}(\alpha)$, they found non-sharp estimates on the first two Taylor–Maclaurin coefficients $|a_2|$ and $|a_3|$ (for details, see (Brannan & Taha, 1988; Taha, 1981)).

Recently, Srivastava et al. (Srivastava et al., 2010), Frasin (Frasin, 2014), Frasin and Aouf (Frasin & Aouf, 2011), Goyal and Goswami (Goyal & P.Goswami, 2012), Li and Wang (Li & Wang, 2012), Siregar and Raman (Siregar & Raman, 2012) and Caglar et al. (Caglar et al., 2012) introduced new subclasses of bi-univalent functions and found estimates on the coefficients $|a_2|$ and $|a_3|$ for functions in these classes.

The object of the present paper is to introduce two new subclasses of the function class Σ and find estimates on the coefficients $|a_2|$ and $|a_3|$ for functions in these new subclasses of the function class Σ .

In order to establish our main results, we shall require the following lemma:

Lemma 1. (Pommerenke, 1975) If $p \in \mathcal{P}$, then $|c_k| \leq 2$ for each k , where \mathcal{P} is the family of all functions p analytic in \mathcal{U} for which

$$\operatorname{Re}(p(z)) > 0, \quad p(z) = 1 + c_1z + c_2z^2 + \dots \quad (z \in \mathcal{U}).$$

2. Coefficient bounds for the function class $\mathcal{L}_\Sigma(\theta, \alpha)$

We now introduce the subclass $\mathcal{L}_\Sigma(\theta, \alpha)$ of the functions in the class \mathcal{A} as follows.

Definition 2.1. A function $f(z)$ given by (1.1) is said to be in the class $\mathcal{L}_\Sigma(\theta, \alpha)$ where $0 < \alpha \leq 1$ and $\theta \in (-\pi, \pi]$, if the following conditions are satisfied:

$$f \in \Sigma \text{ and } \left| \arg \left(f'(z) + \frac{1 + e^{i\theta}}{2} z f''(z) \right) \right| < \frac{\alpha\pi}{2} \quad (z \in \mathcal{U}) \tag{2.1}$$

and

$$\left| \arg \left(g'(w) + \frac{1 + e^{i\theta}}{2} w g''(w) \right) \right| < \frac{\alpha\pi}{2} \quad (w \in \mathcal{U}), \tag{2.2}$$

where the function g is given by

$$g(w) = w - a_2w^2 + (2a_2^2 - a_3)w^3 - (5a_2^3 - 5a_2a_3 + a_4)w^4 + \dots \tag{2.3}$$

We first state and prove the following result.

Theorem 1. Let $f(z)$ given by (1.1) be in the function class $\mathcal{L}_\Sigma(\theta, \alpha)$ where $0 < \alpha \leq 1$ and $\theta \in (-\pi, \pi]$. Then

$$|a_2| \leq \frac{2\alpha}{[(3\alpha + 9 + (1 - \alpha) \cos 2\theta + 6 \cos \theta)^2 + ((1 - \alpha) \sin 2\theta + 6 \sin \theta)^2]^{1/4}} \tag{2.4}$$

and

$$|a_3| \leq \frac{2\alpha^2}{5 + 3 \cos \theta} + \frac{2\alpha}{3\sqrt{5 + 4 \cos \theta}}. \tag{2.5}$$

Proof. It follows from (2.1) and (2.2) that

$$f'(z) + \left(\frac{1 + e^{i\theta}}{2} \right) z f''(z) = [p(z)]^\alpha \tag{2.6}$$

and

$$g'(w) + \left(\frac{1 + e^{i\theta}}{2} \right) w g''(w) = [q(w)]^\alpha \tag{2.7}$$

where $p(z)$ and $q(w)$ are in \mathcal{P} and have the forms

$$p(z) = 1 + p_1z + p_2z^2 + p_3z^3 + \dots \tag{2.8}$$

and

$$q(w) = 1 + q_1w + q_2w^2 + q_3w^3 + \dots \tag{2.9}$$

Now, equating the coefficients in (2.6) and (2.7), we get

$$(3 + e^{i\theta})a_2 = \alpha p_1, \tag{2.10}$$

$$3(2 + e^{i\theta})a_3 = \alpha p_2 + \frac{\alpha(\alpha - 1)}{2} p_1^2, \tag{2.11}$$

$$-(3 + e^{i\theta})a_2 = \alpha q_1 \tag{2.12}$$

and

$$3(2 + e^{i\theta})(2a_2^2 - a_3) = \alpha q_2 + \frac{\alpha(\alpha - 1)}{2} q_1^2. \tag{2.13}$$

From (2.10) and (2.12), we get

$$p_1 = -q_1 \tag{2.14}$$

and

$$2(3 + e^{i\theta})^2 a_2^2 = \alpha^2 (p_1^2 + q_1^2). \tag{2.15}$$

Now from (2.11), (2.13) and (2.15), we obtain

$$\begin{aligned} 6(2 + e^{i\theta})a_2^2 &= \alpha(p_2 + q_2) + \frac{\alpha(\alpha - 1)}{2} (p_1^2 + q_1^2) \\ &= \alpha(p_2 + q_2) + \frac{(\alpha - 1)(3 + e^{i\theta})^2}{\alpha} a_2^2. \end{aligned}$$

Thus

$$a_2^2 = \frac{\alpha^2(p_2 + q_2)}{6\alpha(2 + e^{i\theta}) - (\alpha - 1)(3 + e^{i\theta})^2}$$

that is

$$|a_2^2| = \frac{\alpha^2 |p_2 + q_2|}{|6\alpha(2 + e^{i\theta}) - (\alpha - 1)(3 + e^{i\theta})^2|}$$

Applying Lemma 1 for the coefficients p_2 and q_2 , we have

$$|a_2| \leq \frac{2\alpha}{[(3\alpha + 9 + (1 - \alpha) \cos 2\theta + 6 \cos \theta)^2 + ((1 - \alpha) \sin 2\theta + 6 \sin \theta)^2]^{1/4}}.$$

This gives the bound on $|a_2|$ as asserted in (2.4).

Next, in order to find the bound on $|a_3|$, by subtracting (2.13) from (2.11), we get

$$6(2 + e^{i\theta})a_3 - 6(2 + e^{i\theta})a_2^2 = \alpha p_2 + \frac{\alpha(\alpha - 1)}{2} p_1^2 - \left(\alpha q_2 + \frac{\alpha(\alpha - 1)}{2} q_1^2 \right). \tag{2.16}$$

Upon substituting the value of a_2^2 from (2.15) and observing that $p_1^2 = q_1^2$, it follows that

$$a_3 = \frac{\alpha^2 p_1^2}{(3 + e^{i\theta})^2} + \frac{\alpha(p_2 - q_2)}{6(2 + e^{i\theta})}.$$

Applying Lemma 1 once again for the coefficients p_1, p_2, q_1 and q_2 , we readily get

$$|a_3| \leq \frac{2\alpha^2}{5 + 3 \cos \theta} + \frac{2\alpha}{3 \sqrt{5 + 4 \cos \theta}},$$

which completes the proof of Theorem 1. \square

Choosing $\theta = \pi$ in Theorem 1, we obtain the following particular case due to Srivastava et al. (Srivastava et al., 2010):

Corollary 2.1. (Srivastava et al., 2010) Let $f(z)$ given by (1.1) be in the function class $\mathcal{L}_\Sigma(\pi, \alpha)$; $0 < \alpha \leq 1$. Then

$$|a_2| \leq \alpha \sqrt{\frac{2}{\alpha + 1}} \quad (2.17)$$

and

$$|a_3| \leq \frac{\alpha(3\alpha + 2)}{3}. \quad (2.18)$$

Putting $\theta = 0$ in Theorem 1, we obtain the following particular case due to Frasin (Frasin, 2014):

Corollary 2.2. (Frasin, 2014) Let $f(z)$ given by (1.1) be in the function class $\mathcal{L}_\Sigma(0, \alpha)$, $0 < \alpha \leq 1$. Then

$$|a_2| \leq \alpha \sqrt{\frac{2}{\alpha + 8}} \quad (2.19)$$

and

$$|a_3| \leq \frac{9\alpha^2 + 8\alpha}{36}. \quad (2.20)$$

3. Coefficient bounds for the function class $\mathcal{L}_\Sigma(\theta, \gamma)$

Definition 3.1. A function $f(z)$ given by (1.1) is said to be in the class $\mathcal{L}_\Sigma(\theta, \gamma)$ where $0 \leq \gamma < 1$, $\theta \in (-\pi, \pi]$, if the following conditions are satisfied:

$$f \in \Sigma \text{ and } \operatorname{Re} \left(f'(z) + \frac{1 + e^{i\theta}}{2} z f''(z) \right) > \gamma \quad (z \in \mathcal{U}) \quad (3.1)$$

and

$$\operatorname{Re} \left(g'(w) + \frac{1 + e^{i\theta}}{2} w g''(w) \right) > \gamma \quad (w \in \mathcal{U}), \quad (3.2)$$

where the function g is given by (2.3).

Theorem 2. Let $f(z)$ given by (1.1) be in the class $\mathcal{L}_\Sigma(\theta, \gamma)$, where $0 \leq \gamma < 1, \theta \in (-\pi, \pi]$. Then

$$|a_2| \leq \sqrt{\frac{4(1-\gamma)}{6\sqrt{5+4\cos\theta}}} \tag{3.3}$$

and

$$|a_3| \leq \frac{2(1-\gamma)^2}{5+3\cos\theta} + \frac{2(1-\gamma)}{3\sqrt{5+4\cos\theta}}. \tag{3.4}$$

Proof. It follows from (3.1) and (3.2) that there exist p and $q \in \mathcal{P}$ such that

$$f'(z) + \left(\frac{1+e^{i\theta}}{2}\right)zf''(z) = \gamma + (1-\gamma)p(z) \tag{3.5}$$

and

$$g'(w) + \left(\frac{1+e^{i\theta}}{2}\right)wg''(w) = \gamma + (1-\gamma)q(w) \tag{3.6}$$

where $p(z)$ and $q(w)$ have the forms (2.8) and (2.9), respectively. Equating coefficients in (3.5) and (3.6) yields

$$(3+e^{i\theta})a_2 = (1-\gamma)p_1, \tag{3.7}$$

$$3(2+e^{i\theta})a_3 = (1-\gamma)p_2, \tag{3.8}$$

$$-(3+e^{i\theta})a_2 = (1-\gamma)q_1 \tag{3.9}$$

and

$$3(2+e^{i\theta})(2a_2^2 - a_3) = (1-\gamma)q_2 \tag{3.10}$$

From (3.7) and (3.9), we get

$$p_1 = -q_1 \tag{3.11}$$

and

$$2(3+e^{i\theta})^2a_2^2 = (1-\gamma)^2(p_1^2 + q_1^2). \tag{3.12}$$

Also, from (3.8) and (3.10), we find that

$$6(2+e^{i\theta})a_2^2 = (1-\gamma)(p_2 + q_2).$$

Thus, we have

$$\begin{aligned} |a_2^2| &\leq \frac{(1-\gamma)}{6|2+e^{i\theta}|}(|p_2| + |q_2|) \\ &\leq \frac{4(1-\gamma)}{6\sqrt{5+4\cos\theta}} \end{aligned}$$

which is the bound on $|a_2|$ as given in (3.3).

Next, in order to find the bound on $|a_3|$, by subtracting (3.10) from (3.8), we get

$$6(2 + e^{i\theta})a_3 - 6(2 + e^{i\theta})a_2^2 = (1 - \gamma)(p_2 - q_2)$$

or, equivalently,

$$a_3 = a_2^2 + \frac{(1 - \gamma)(p_2 - q_2)}{6(2 + e^{i\theta})}.$$

Upon substituting the value of a_2^2 from (3.12), we obtain

$$a_3 = \frac{(1 - \gamma)^2(p_1^2 + q_1^2)}{2(3 + e^{i\theta})^2} + \frac{(1 - \gamma)(p_2 - q_2)}{6(2 + e^{i\theta})}.$$

Applying Lemma 1 for the coefficients p_1, p_2, q_1 and q_2 , we readily get

$$|a_3| \leq \frac{2(1 - \gamma)^2}{5 + 3 \cos \theta} + \frac{2(1 - \gamma)}{3 \sqrt{5 + 4 \cos \theta}}$$

which is the bound on $|a_3|$ as asserted in (3.4). □

Choosing $\theta = \pi$ in Theorem 2, we obtain the following particular case due to Srivastava et al. (Srivastava et al., 2010):

Corollary 3.1. (Srivastava et al., 2010) Let $f(z)$ given by (1.1) be in the function class $\mathcal{L}_\Sigma(0, \gamma)$, $0 \leq \gamma < 1$. Then

$$|a_2| \leq \sqrt{\frac{2(1 - \gamma)}{3}} \tag{3.13}$$

and

$$|a_3| \leq \frac{(1 - \gamma)(5 - 3\gamma)}{3}. \tag{3.14}$$

Putting $\theta = 0$ in Theorem 2, we obtain the following particular case due to Frasin (Frasin, 2014):

Corollary 3.2. (Frasin, 2014) Let $f(z)$ given by (1.1) be in the function class $\mathcal{L}_\Sigma(0, \gamma)$, $0 \leq \gamma < 1$. Then

$$|a_2| \leq \frac{1}{3} \sqrt{2(1 - \gamma)} \tag{3.15}$$

and

$$|a_3| \leq \frac{(1 - \gamma)(9(1 - \gamma) + 8)}{36}. \tag{3.16}$$

Acknowledgments

The authors thanks the referee for his valuable suggestions which led to improvement of this study.

References

- Alexander, J.W. (1915). Functions which map the interior of the unit circle upon simple regions. *Annals of Mathematics* **17**, 12–22.
- Brannan, D.A. and T.S. Taha (1988). On some classes of bi-univalent functions. In: *KFAS Proceedings Series*. Vol. 3. Pergamon Press, Elsevier Science Limited, Oxford. pp. 53–60.
- Caglar, M., H. Orhan and N. Yagmur (2012). Coefficient bounds for new subclasses of bi-univalent functions. *arXiv:1204.4285* pp. 1–7.
- Chichra, P.N. (1977). New subclasses of the class of close-to-convex functions. *Proc. Amer. Math. Soc.* **62**, 37–43.
- Frasin, B.A. (2014). Coefficient bounds for certain classes of bi-univalent functions. *Hacettepe Journal of Mathematics and Statistics* **43**(3), 383–389.
- Frasin, B.A. and M.K. Aouf (2011). New subclasses of bi-univalent functions. *Applied Mathematics Letters* **24**(9), 1569–1573.
- Goyal, S.P. and P.Goswami (2012). Estimate for initial maclaurin coefficients of bi-univalent functions for a class defined by fractional derivatives. *Journal of the Egyptian Mathematical Society* **20**, 179–182.
- Lewandowski, Z., S.S Miller and E. Zlotkiewicz (1976). Generating functions for some classes of univalent functions. *Proc. Amer. Math. Soc.* **56**, 111–117.
- Li, X.F. and A.P. Wang (2012). Two new subclasses of bi-univalent functions. *International Mathematical Forum* **7**(30), 1495–1504.
- Pommerenke, Ch. (1975). *Univalent functions*. Vandenhoeck and Ruprecht, Gottingen.
- Silverman, H.S. (1994). A class of bounded starlike functions. *Internat. J. Math. Sci.* **17**(2), 249–252.
- Silverman, H.S. and E.M. Silvia (1999). Characterizations for subclasses of univalent functions. *Math. Japonica* **50**(1), 103–109.
- Singh, R. and S. Singh (1989). Convolution properties of a class of starlike functions. *Proc. Amer. Math. Soc.* **106**, 145–152.
- Siregar, S. and S. Raman (2012). Certain subclasses of analytic and bi-univalent functions involving double zeta functions. *International Journal of Advanced on Science Engineering Information Technology* **2**(5), 16–18.
- Srivastava, H.M., A.K. Mishra and P. Gochhayat (2010). Certain subclasses of analytic and bi-univalent functions. *Applied Mathematics Letters* **23**(5), 1188–1192.
- Taha, T.S. (1981). *Topics in Univalent Function Theory*. Ph.D. Thesis, University of London.



Zweier I-Convergent Double Sequence Spaces Defined by a Sequence of Modulii

Vakeel A. Khan^{a,*}, Nazneen Khan^a, Yasmeen^a

^aDepartment of Mathematics, Aligarh Muslim University, Aligarh-202002, India

Abstract

In the present article we have introduced the double sequence spaces ${}_2\mathcal{Z}^I(F)$, ${}_2\mathcal{Z}_0^I(F)$ and ${}_2\mathcal{Z}_\infty^I(F)$ for a sequence of modulii $F = (f_{ij})$. We have also studied their topological as well as algebraic properties.

Keywords: Ideal, filter, double sequence, sequence of modulii, Lipschitz function, I-convergence, monotone and solid spaces.

2010 MSC: 30C45, 30C50.

1. Introduction

Let \mathbb{N} , \mathbb{R} and \mathbb{C} be the sets of all natural, real and complex numbers respectively. We write

$$\omega = \{x = (x_k) : x_k \in \mathbb{R} \text{ or } \mathbb{C}\}$$

the space of all real or complex sequences. Let ℓ_∞ , c and c_0 denote the Banach spaces of bounded, convergent and null sequences respectively normed by $\|x\|_\infty = \sup_k |x_k|$.

The concept of statistical convergence was first introduced by Fast (Fast, 1951) in 1951 and also independently by Buck (Buck, 1953) and Schoenberg (Schoenberg, 1959) for real and complex sequences. Further this concept was studied by Connor (Connor, 1998, 1989; Connor & Kline, 1996), Connor, Fridy and Kline (Connor *et al.*, 1994) and many others. Statistical convergence is a generalization of the usual notion of convergence that parallels the usual theory of convergence. A sequence $x = (x_k)$ is said to be Statistically convergent to L if for a given $\epsilon > 0$

$$\lim_k \frac{1}{k} |\{i : |x_i - L| \geq \epsilon, i \leq k\}| = 0.$$

*Corresponding author

Email addresses: vakhanmaths@gmail.com (Vakeel A. Khan), nazneen4maths@gmail.com (Nazneen Khan)

Later on it was studied by Fridy (Fridy, 1985, 1993) from the sequence space point of view and he linked it with the summability theory. The notion of I-convergence is a generalization of the statistical convergence. At the initial stage it was studied by Kostyrko, Šalát, Wilczyński (Kostyrko et al., 2000). Later on it was studied by Šalát, Tripathy, Ziman (Šalát et al., 2004) and Demirci (Demirci, 2001).

Here we give some preliminaries about the notion of I-convergence.

Let X be a non empty set. A set $I \subseteq 2^X$ (2^X denoting the power set of X) is said to be an ideal if I is additive i.e $A, B \in I \Rightarrow A \cup B \in I$ and hereditary i.e $A \in I, B \subseteq A \Rightarrow B \in I$. A non empty family of sets $\mathcal{F} \subseteq 2^X$ is said to be filter on X if and only if $\emptyset \notin \mathcal{F}$, for $A, B \in \mathcal{F}$ we have $A \cap B \in \mathcal{F}$ and for each $A \in \mathcal{F}$ and $A \subseteq B$ implies $B \in \mathcal{F}$.

An Ideal $I \subseteq 2^X$ is called non-trivial if $I \neq 2^X$. A non-trivial ideal $I \subseteq 2^X$ is called admissible if $\{\{x\} : x \in X\} \subseteq I$. A non-trivial ideal I is maximal if there cannot exist any non-trivial ideal $J \neq I$ containing I as a subset. For each ideal I , there is a filter $\mathcal{F}(I)$ corresponding to I , i.e $\mathcal{F}(I) = \{K \subseteq \mathbb{N} : K^c \in I\}$, where $K^c = \mathbb{N} - K$.

Each linear subspace of ω , for example, $\lambda, \mu \subset \omega$ is called a sequence space. A sequence space λ with linear topology is called a K -space provided each of maps $p_i \rightarrow \mathbb{C}$ defined by $p_i(x) = x_i$ is continuous for all $i \in \mathbb{N}$. A K -space λ is called an FK-space provided λ is a complete linear metric space. An FK-space whose topology is normable is called a BK-space. Let λ and μ be two sequence spaces and $A = (a_{nk})$ is an infinite matrix of real or complex numbers a_{nk} , where $n, k \in \mathbb{N}$. Then we say that A defines a matrix mapping from λ to μ and we denote it by writing $A : \lambda \rightarrow \mu$.

If for every sequence $x = (x_k) \in \lambda$ the sequence $Ax = \{(Ax)_n\}$, the A transform of x is in μ , where

$$(Ax)_n = \sum_k a_{nk}x_k, \quad (n \in \mathbb{N}). \quad (1.1)$$

By $(\lambda : \mu)$, we denote the class of matrices A such that $A : \lambda \rightarrow \mu$.

Thus, $A \in (\lambda : \mu)$ if and only if series on the right side of (1.1) converges for each $n \in \mathbb{N}$ and every $x \in \lambda$. The approach of constructing the new sequence spaces by means of the matrix domain of a particular limitation method have been recently employed by Altay, Başar and Mursaleen (Altay et al., 2006), Başar and Altay (Başar & Altay, 2003), Malkowsky (Malkowsky, 1997), Ng and Lee (Ng & Lee, 1978) and Wang (Wang, 1978). Şengönül (Şengönül, 2007) defined the sequence $y = (y_i)$ which is frequently used as the Z^p transform of the sequence $x = (x_i)$ i.e,

$$y_i = px_i + (1 - p)x_{i-1}$$

where $x_{-1} = 0, p \neq 1, 1 < p < \infty$ and Z^p denotes the matrix $Z^p = (z_{ik})$ defined by

$$z_{ik} = \begin{cases} p, (i = k), \\ 1 - p, (i - 1 = k); (i, k \in \mathbb{N}), \\ 0, \text{otherwise.} \end{cases}$$

Following Basar and Altay (Başar & Altay, 2003), Şengönül Şengönül (2007) introduced the Zweier sequence spaces \mathcal{Z} and \mathcal{Z}_0 as follows

$$\mathcal{Z} = \{x = (x_k) \in \omega : Z^p x \in c\}$$

$$\mathcal{Z}_0 = \{x = (x_k) \in \omega : Z^p x \in c_0\}$$

Definition 1.1. (Khan & Khan, 2013) A double sequence of complex numbers is defined as a function $x : \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{C}$. We denote a double sequence as (x_{ij}) , where the two subscripts run through the sequence of natural numbers independent of each other. A number $a \in \mathbb{C}$ is called a double limit of a double sequence (x_{ij}) if for every $\epsilon > 0$ there exists some $N = N(\epsilon) \in \mathbb{N}$ such that

$$|(x_{ij}) - a| < \epsilon, \quad \text{for all } i, j \geq N$$

Definition 1.2. A double sequence space E is said to be solid or normal if $(x_{ij}) \in E$ implies $(\alpha_{ij}x_{ij}) \in E$ for all sequence of scalars (α_{ij}) with $|\alpha_{ij}| < 1$ for all $(i, j) \in \mathbb{N} \times \mathbb{N}$.

Definition 1.3. E is said to be monotone if it contains the canonical preimages of all its stepspace.

Definition 1.4. E is said to be convergence free if $(y_{ij}) \in E$ whenever $(x_{ij}) \in E$ and $x_{ij} = 0$ implies $y_{ij} = 0$.

Definition 1.5. E is said to be a sequence algebra if $(x_{ij}y_{ij}) \in E$ whenever $(x_{ij}), (y_{ij}) \in E$.

Definition 1.6. A sequence $(x_k) \in \omega$ is said to be I-convergent to a number L if for every $\epsilon > 0$. $\{(i, j) \in \mathbb{N} \times \mathbb{N} : |x_{ij} - L| \geq \epsilon\} \in I$. In this case we write $I\text{-lim } x_{ij} = L$. The space ${}_2c^I$ of all I-convergent double sequences to L is given by

$${}_2c^I = \{(x_k) \in \omega : \{(i, j) \in \mathbb{N} \times \mathbb{N} : |x_{ij} - L| \geq \epsilon\} \in I, \text{ for some } L \in \mathbb{C}\}.$$

Definition 1.7. A sequence $(x_{ij}) \in \omega$ is said to be I-null if $L = 0$. In this case we write $I\text{-lim } x_k = 0$.

Definition 1.8. A sequence $(x_{ij}) \in \omega$ is said to be I-cauchy if for every $\epsilon > 0$ there exists a number m, n dependent on ϵ such that $\{(i, j) \in \mathbb{N} \times \mathbb{N} : |x_{ij} - x_{mn}| \geq \epsilon\} \in I$.

Definition 1.9. A sequence $(x_{ij}) \in \omega$ is said to be I-bounded if there exists $M > 0$ such that $\{(i, j) \in \mathbb{N} \times \mathbb{N} : |x_{ij}| > M\} \in I$.

Definition 1.10. A modulus function f is said to satisfy Δ_2 condition if for all values of u there exists a constant $K > 0$ such that $f(Lu) \leq KLf(u)$ for all values of $L > 1$.

Definition 1.11. Take for I the class I_f of all finite subsets of \mathbb{N} . Then I_f is a non-trivial admissible ideal and I_f convergence coincides with the usual convergence with respect to the metric in X .

Definition 1.12. For $I = I_\delta$ and $A \subset \mathbb{N} \times \mathbb{N}$ with $\delta(A) = 0$ respectively. I_δ is a non-trivial admissible ideal, I_δ -convergence is said to be logarithmic statistical convergence.

Definition 1.13. A map \bar{h} defined on a domain $D \subset X$ i.e $\bar{h} : D \subset X \rightarrow \mathbb{R}$ is said to satisfy Lipschitz condition if $|\bar{h}(x) - \bar{h}(y)| \leq K|x - y|$ where K is known as the Lipschitz constant. The class of K -Lipschitz functions defined on D is denoted by $\bar{h} \in (D, K)$.

Definition 1.14. A convergence field of I-convergence is a set

$$F(I) = \{x = (x_{ij}) \in {}_2\ell_\infty : \text{there exists } I\text{-lim } x \in \mathbb{R}\}.$$

The convergence field $F(I)$ is a closed linear subspace of ${}_2\ell_\infty$ with respect to the supremum norm, $F(I) = {}_2\ell_\infty \cap {}_2c^I$. Define a function $\bar{h} : F(I) \rightarrow \mathbb{R}$ such that $\bar{h}(x) = I - \lim x$, for all $x \in F(I)$, then the function $\bar{h} : F(I) \rightarrow \mathbb{R}$ is a Lipschitz function. The following Lemmas will be used for establishing some results of this article.

Lemma 1. *Let E be a sequence space. If E is solid then E is monotone.*

Lemma 2. *Let $K \in \mathfrak{L}(I)$ and $M \subseteq N$. If $M \notin I$, then $M \cap N \notin I$. (Tripathy & Hazarika, 2011).*

Lemma 3. *If $I \subset 2^N$ and $M \subseteq N$. If $M \notin I$, then $M \cap N \notin I$. (Tripathy & Hazarika, 2011).*

The idea of modulus was structured in 1953 by Nakano (Nakano, 1953).

A function $f : [0, \infty) \rightarrow [0, \infty)$ is called a modulus if

- (1) $f(t) = 0$ if and only if $t = 0$,
- (2) $f(t + u) \leq f(t) + f(u)$ for all $t, u \geq 0$,
- (3) f is nondecreasing, and
- (4) f is continuous from the right at zero.

Ruckle (Ruckle, 1968, 1967) used the idea of a modulus function f to construct the sequence space

$$X(f) = \{x = (x_k) : \sum_{k=1}^{\infty} f(|x_k|) < \infty\}.$$

This space is an FK space, and Ruckle (Ruckle, 1973) proved that the intersection of all such $X(f)$ spaces is ϕ , the space of all finite sequences. The space $X(f)$ is closely related to the space ℓ_1 which is an $X(f)$ space with $f(x) = x$ for all real $x \geq 0$. Thus Ruckle (Ruckle, 1973) proved that, for any modulus f :

$$X(f) \subset \ell_1 \text{ and } X(f)^\alpha = \ell_\infty$$

where

$$X(f)^\alpha = \{y = (y_k) \in \omega : \sum_{k=1}^{\infty} f(|y_k x_k|) < \infty\}.$$

The space $X(f)$ is a Banach space with respect to the norm

$$\|x\| = \sum_{k=1}^{\infty} f(|x_k|) < \infty.$$

From the point of view of local convexity, spaces of the type $X(f)$ are quite pathological. Therefore symmetric sequence spaces, which are locally convex have been frequently studied by Garling (Garling, 1966), Köthe (Köthe, 1970) and many more. After then Kolk (Kolk, 1993, 1994) gave an extension of $X(f)$ by considering a sequence of moduli $F = (f_k)$ and defined the sequence space

$$X(F) = \{x = (x_k) : (f_k(|x_k|)) \in X\}.$$

Recently Khan et al (Khan et al., 2013), introduced the following classes of sequences

$$\mathcal{Z}^I(f) = \{(x_k) \in \omega : \{k \in \mathbb{N} : f(|x_k - L|) \geq \varepsilon, \text{ for some } L \in \mathbb{C}\} \in I\},$$

$$\mathcal{Z}_0^I(f) = \{(x_k) \in \omega : \{k \in \mathbb{N} : f(|x_k|) \geq \varepsilon\} \in I\},$$

$$\mathcal{Z}_\infty^I(f) = \{(x_k) \in \omega : \{k \in \mathbb{N} : f(|x_k|) \geq M, \text{ for each fixed } M > 0\} \in I\}.$$

We also denote by

$$m_{\mathcal{Z}}^I(f) = \mathcal{Z}_\infty^I(f) \cap \mathcal{Z}^I(f)$$

and

$$m_{\mathcal{Z}_0}^I(f) = \mathcal{Z}_\infty^I(f) \cap \mathcal{Z}_0^I(f).$$

In this article we introduce the following sequence spaces.

$${}_2\mathcal{Z}^I(F) = \{(x_{ij}) \in {}_2\omega : \{(i, j) \in \mathbb{N} \times \mathbb{N} : f_{ij}(|x_{ij} - L|) \geq \varepsilon, \text{ for some } L \in \mathbb{C}\} \in I\},$$

$${}_2\mathcal{Z}_0^I(F) = \{(x_{ij}) \in {}_2\omega : \{(i, j) \in \mathbb{N} \times \mathbb{N} : f_{ij}(|x_{ij}|) \geq \varepsilon\} \in I\},$$

$${}_2\mathcal{Z}_\infty^I(F) = \{(x_{ij}) \in {}_2\omega : \{(i, j) \in \mathbb{N} \times \mathbb{N} : f_{ij}(|x_{ij}|) \geq M, \text{ for each fixed } M > 0\} \in I\}.$$

We also denote by

$${}_2m_{\mathcal{Z}}^I(F) = {}_2\mathcal{Z}_\infty^I(F) \cap {}_2\mathcal{Z}^I(F)$$

and

$${}_2m_{\mathcal{Z}_0}^I(F) = {}_2\mathcal{Z}_\infty^I(F) \cap {}_2\mathcal{Z}_0^I(F).$$

2. Main Results

Theorem 1. For a sequence of moduli $F = (f_{ij})$, the classes of sequences ${}_2\mathcal{Z}^I(F)$, ${}_2\mathcal{Z}_0^I(F)$, ${}_2m_{\mathcal{Z}}^I(F)$ and ${}_2m_{\mathcal{Z}_0}^I(F)$ are linear spaces.

Proof. We shall prove the result for the space ${}_2\mathcal{Z}^I(F)$. The proof for the other spaces will follow similarly. Let $(x_{ij}), (y_{ij}) \in {}_2\mathcal{Z}^I(F)$ and let α, β be scalars. Then

$$I - \lim f_{ij}(|x_{ij} - L_1|) = 0, \text{ for some } L_1 \in \mathbb{C};$$

$$I - \lim f_{ij}(|y_{ij} - L_2|) = 0, \text{ for some } L_2 \in \mathbb{C};$$

That is for a given $\epsilon > 0$, we have

$$A_1 = \{(i, j) \in \mathbb{N} \times \mathbb{N} : f_{ij}(|x_{ij} - L_1|) > \frac{\epsilon}{2}\} \in I, \tag{2.1}$$

$$A_2 = \{(i, j) \in \mathbb{N} \times \mathbb{N} : f_{ij}(|y_{ij} - L_2|) > \frac{\epsilon}{2}\} \in I. \tag{2.2}$$

Since f_{ij} is a modulus function, we have

$$\begin{aligned} f_{ij}(|(\alpha x_{ij} + \beta y_{ij}) - (\alpha L_1 + \beta L_2)|) &\leq f_{ij}(|\alpha||x_{ij} - L_1|) + f_{ij}(|\beta||y_{ij} - L_2|) \\ &\leq f_{ij}(|x_{ij} - L_1|) + f_{ij}(|y_{ij} - L_2|) \end{aligned}$$

Now, by (2.1) and (2.2), $\{(i, j) \in \mathbb{N} \times \mathbb{N} : f_{ij}(|(\alpha x_{ij} + \beta y_{ij}) - (\alpha L_1 + \beta L_2)|) > \epsilon\} \subset A_1 \cup A_2$. Therefore $(\alpha x_{ij} + \beta y_{ij}) \in {}_2\mathcal{Z}^I(F)$. Hence ${}_2\mathcal{Z}^I(F)$ is a linear space. □

We state the following result without proof in view of Theorem 2.1.

Theorem 2. The spaces ${}_2m_Z^I(F)$ and ${}_2m_{Z_0}^I(F)$ are normed linear spaces, normed by

$$\|x_{ij}\|_* = \sup_{i,j} f_{ij}(|x_{ij}|). \tag{2.3}$$

Theorem 3. A sequence $x = (x_{ij}) \in {}_2m_Z^I(F)$ I-converges if and only if for every $\epsilon > 0$ there exists $N_\epsilon \in \mathbb{N} \times \mathbb{N}$ such that

$$\{(i, j) \in \mathbb{N} \times \mathbb{N} : f_{ij}(|x_{ij} - x_{N_\epsilon}|) < \epsilon\} \in \mathcal{F}(I). \tag{2.4}$$

Proof. Suppose that $L = I - \lim x$. Then $B_\epsilon = \{(i, j) \in \mathbb{N} \times \mathbb{N} : |x_{ij} - L| < \frac{\epsilon}{2}\} \in \mathcal{F}(I)$. For all $\epsilon > 0$, fix an $N_\epsilon \in B_\epsilon$ such that we have $|x_{N_\epsilon} - x_{ij}| \leq |x_{N_\epsilon} - L| + |L - x_{ij}| < \frac{\epsilon}{2} + \frac{\epsilon}{2} = \epsilon$ which holds for all $(i, j) \in B_\epsilon$. Hence $\{(i, j) \in \mathbb{N} \times \mathbb{N} : f_{ij}(|x_{ij} - x_{N_\epsilon}|) < \epsilon\} \in \mathcal{F}(I)$.

Conversely, suppose that $\{(i, j) \in \mathbb{N} \times \mathbb{N} : f_{ij}(|x_{ij} - x_{N_\epsilon}|) < \epsilon\} \in \mathcal{F}(I)$. That is $\{(i, j) \in \mathbb{N} \times \mathbb{N} : (|x_{ij} - x_{N_\epsilon}|) < \epsilon\} \in \mathcal{F}(I)$ for all $\epsilon > 0$. Then the set $C_\epsilon = \{(i, j) \in \mathbb{N} \times \mathbb{N} : x_{ij} \in [x_{N_\epsilon} - \epsilon, x_{N_\epsilon} + \epsilon]\} \in \mathcal{F}(I)$ for all $\epsilon > 0$.

Let $J_\epsilon = [x_{N_\epsilon} - \epsilon, x_{N_\epsilon} + \epsilon]$. If we fix an $\epsilon > 0$ then we have $C_\epsilon \in \mathcal{F}(I)$ as well as $C_{\frac{\epsilon}{2}} \in \mathcal{F}(I)$.

Hence $C_\epsilon \cap C_{\frac{\epsilon}{2}} \in {}_2m_Z^I(F)$. This implies that $J_\epsilon \cap J_{\frac{\epsilon}{2}} \neq \emptyset$ that is $\{(i, j) \in \mathbb{N} \times \mathbb{N} : x_{ij} \in J\} \in \mathcal{F}(I)$ that is $\text{diam}J \leq \text{diam}J_\epsilon$ where the diam of J denotes the length of interval J .

In this way, by induction we get the sequence of closed intervals $J_\epsilon = I_0 \supseteq I_1 \supseteq \dots \supseteq I_{ij} \supseteq \dots$ with the property that $\text{diam}I_{ij} \leq \frac{1}{2}\text{diam}I_{(i-1)(j-1)}$ for $(i,j)=2,3,4, \dots$ and $\{(i, j) \in \mathbb{N} \times \mathbb{N} : x_{ij} \in {}_2m_Z^I(F)\} \in I_{ij}$ for $(i,j)=1,2,3,4, \dots$. Then there exists a $\xi \in \bigcap I_{ij}$ where $(i, j) \in \mathbb{N} \times \mathbb{N}$ such that $\xi = I - \lim x$. So that $f_{ij}(\xi) = I - \lim f_{ij}(x)$, that is $L = I - \lim f_{ij}(x)$. □

Theorem 4. Let (f_{ij}) and (g_{ij}) be modulus functions for some fixed k that satisfy the Δ_2 -condition. If X is any of the spaces ${}_2Z^I, {}_2Z_0^I, {}_2m_Z^I$ and ${}_2m_{Z_0}^I$ etc., then the following assertions hold.

- (a) $X(g_{ij}) \subseteq X(f_{ij} \cdot g_{ij})$,
- (b) $X(f_{ij}) \cap X(g_{ij}) \subseteq X(f_{ij} + g_{ij})$

Proof. (a) Let $(x_{mn}) \in {}_2Z_0^I(g_{ij})$. Then

$$I - \lim_{m,n} g_{ij}(|x_{mn}|) = 0. \tag{2.5}$$

Let $\epsilon > 0$ and choose δ with $0 < \delta < 1$ such that $f_{ij}(t) < \epsilon$ for $0 < t < \delta$. Write $y_{mn} = g_{ij}(|x_{mn}|)$ and consider

$$\lim_{m,n} f_{ij}(y_{mn}) = \lim_{m,n} f_{ij}(y_{mn})_{y_{mn} < \delta} + \lim_{m,n} f_{ij}(y_{mn})_{y_{mn} \geq \delta} \tag{2.6}$$

Now for $y_{mn} < \delta$, we already have $\lim_{m,n} f_{ij}(y_{mn}) < \epsilon$. For $y_{mn} \geq \delta$, we have $y_{mn} < \frac{y_{mn}}{\delta} < 1 + \frac{y_{mn}}{\delta}$.

Since f_{ij} is non-decreasing, it follows that $f_{ij}(y_{mn}) < f_{ij}(1 + \frac{y_{mn}}{\delta}) < \frac{1}{2}f_{ij}(2) + \frac{1}{2}f_{ij}(\frac{2y_{mn}}{\delta})$

Since f_{ij} satisfies the Δ_2 -condition, therefore for $y_{mn} \geq \delta > 0$ we can choose some $K > 0$ such that $f_{ij}(y_{mn}) < \frac{1}{2}K\frac{y_{mn}}{\delta} f_{ij}(2) + \frac{1}{2}K\frac{y_{mn}}{\delta} f_{ij}(2) = K\frac{y_{mn}}{\delta} f_{ij}(2)$

Hence $\lim_{m,n} f_{ij}(y_{mn}) \leq \max(1, K)\delta^{-1} f_{ij}(2) \lim_{m,n}(y_{mn}) = \epsilon'$ (say). Substituting in equation (2.6), we get

$$\lim_{m,n} f_{ij}(y_{mn}) = \epsilon + \epsilon'. \tag{2.7}$$

From (2.5), (2.6) and (2.7), we have $I - \lim_{m,n} f_{ij}(g_{ij}(|x_{mn}|)) = 0$.

Hence $(x_{mn}) \in {}_2\mathcal{Z}_0^I(f_{ij} \cdot g_{ij})$. Thus ${}_2\mathcal{Z}_0^I(g_{ij}) \subseteq {}_2\mathcal{Z}_0^I(f_{ij} \cdot g_{ij})$. The other cases can be proved similarly.

(b) Let $(x_{mn}) \in {}_2\mathcal{Z}_0^I(f_{ij}) \cap {}_2\mathcal{Z}_0^I(g_{ij})$. Then $I - \lim_{m,n} f_{ij}(|x_{mn}|) = 0$ and $I - \lim_{m,n} g_{ij}(|x_{mn}|) = 0$

The rest of the proof follows from the following equality $\lim_{m,n} (f_{ij} + g_{ij})(|x_{mn}|) = \lim_{m,n} f_{ij}(|x_{mn}|) + \lim_{m,n} g_{ij}(|x_{mn}|)$. □

Corollary 2.1. $X \subseteq X(f_{ij})$ for some fixed (i, j) and $X = {}_2\mathcal{Z}^I, {}_2\mathcal{Z}_0^I, {}_2m_{\mathcal{Z}}^I$ and ${}_2m_{\mathcal{Z}_0}^I$.

Theorem 5. The spaces ${}_2\mathcal{Z}_0^I(F)$ and ${}_2m_{\mathcal{Z}_0}^I(F)$ are solid and monotone .

Proof. We shall prove the result for ${}_2\mathcal{Z}_0^I(F)$.

Let $(x_{ij}) \in {}_2\mathcal{Z}_0^I(F)$. Then

$$I - \lim_{(i,j)} f_{ij}(|x_{ij}|) = 0. \tag{2.8}$$

Let (α_{ij}) be a sequence of scalars with $|\alpha_{ij}| \leq 1$ for all $(i, j) \in \mathbb{N} \times \mathbb{N}$. Then the result follows from (2.8) and inequality $f_{ij}(|\alpha_{ij}x_{ij}|) \leq |\alpha_{ij}|f_{ij}(|x_{ij}|) \leq f_{ij}(|x_{ij}|)$ for all $(i, j) \in \mathbb{N} \times \mathbb{N}$. The space ${}_2\mathcal{Z}_0^I(F)$ is monotone follows from Lemma 1. For ${}_2m_{\mathcal{Z}_0}^I(F)$ the result can be proved similarly. □

Theorem 6. The spaces ${}_2\mathcal{Z}^I(F)$ and ${}_2\mathcal{Z}_0^I(F)$ are sequence algebras.

Proof. We prove the result for ${}_2\mathcal{Z}_0^I(F)$. Let $(x_{ij}), (y_{ij}) \in {}_2\mathcal{Z}_0^I(F)$. Then

$$I - \lim f_{ij}(|x_{ij}|) = 0$$

and

$$I - \lim f_{ij}(|y_{ij}|) = 0$$

Therefore, we have

$$I - \lim f_{ij}(|(x_{ij} \cdot y_{ij})|) = 0.$$

Thus $(x_{ij} \cdot y_{ij}) \in {}_2\mathcal{Z}_0^I(F)$ and hence ${}_2\mathcal{Z}_0^I(F)$ is a sequence algebra. In a similar way we can prove the result for the space ${}_2\mathcal{Z}^I(F)$. □

Theorem 7. The spaces ${}_2\mathcal{Z}^I(F)$ and ${}_2\mathcal{Z}_0^I(F)$ are not convergence free in general.

Proof. Here we give a counter example. Let $I = I_f$ and $f_{ij}(x) = x^3$ for some fixed (i, j) and for all $x = (x_{mn}) \in [0, \infty)$. Consider the sequence (x_{mn}) and (y_{mn}) defined by

$$x_{mn} = \frac{1}{m+n} \quad \text{and} \quad y_{mn} = m+n \quad \text{for all } (m, n) \in \mathbb{N} \times \mathbb{N}.$$

Then $(x_{mn}) \in {}_2\mathcal{Z}^I(F) \cap {}_2\mathcal{Z}_0^I(F)$, but $(y_{mn}) \notin {}_2\mathcal{Z}^I(F) \cap {}_2\mathcal{Z}_0^I(F)$. Hence the spaces ${}_2\mathcal{Z}_0^I(F)$ and ${}_2\mathcal{Z}^I(F)$ are not convergence free. □

Theorem 8. ${}_2\mathcal{Z}_0^I(F) \subset {}_2\mathcal{Z}^I(F) \subset {}_2\mathcal{Z}_\infty^I(F)$.

Proof. Let $(x_{ij}) \in {}_2\mathcal{Z}^I(F)$. Then there exists $L \in \mathbb{C}$ such that $I - \lim f_{ij}(|x_{ij} - L|) = 0$. We have

$$f_{ij}(|x_{ij}|) \leq \frac{1}{2}f_{ij}(|x_{ij} - L|) + \frac{1}{2}f_{ij}(|L|).$$

Taking the supremum over (i, j) on both sides we get $(x_{ij}) \in {}_2\mathcal{Z}^I_\infty(F)$. The inclusion ${}_2\mathcal{Z}^I_0(F) \subset {}_2\mathcal{Z}^I(F)$ is obvious. \square

Theorem 9. *The function $\hbar : {}_2m^I_{\mathcal{Z}}(F) \rightarrow \mathbb{R}$ is the Lipschitz function, where ${}_2m^I_{\mathcal{Z}}(F) = {}_2\mathcal{Z}^I_\infty(F) \cap {}_2\mathcal{Z}^I(F)$, and hence uniformly continuous.*

Proof. Let $x = (x_{ij}), y = (y_{ij}) \in {}_2m^I_{\mathcal{Z}}(F), x \neq y$.

Then the sets

$$A_x = \{(i, j) \in \mathbb{N} \times \mathbb{N} : |x_{ij} - \hbar(x)| \geq \|x - y\|_*\} \in I,$$

$$A_y = \{(i, j) \in \mathbb{N} \times \mathbb{N} : |y_{ij} - \hbar(y)| \geq \|x - y\|_*\} \in I.$$

Thus the sets,

$$B_x = \{(i, j) \in \mathbb{N} \times \mathbb{N} : |x_{ij} - \hbar(x)| < \|x - y\|_*\} \in \mathcal{F}(I),$$

$$B_y = \{(i, j) \in \mathbb{N} \times \mathbb{N} : |y_{ij} - \hbar(y)| < \|x - y\|_*\} \in \mathcal{F}(I).$$

Hence also $B = B_x \cap B_y \in m^I_{\mathcal{Z}}(F)$, so that $B \neq \phi$.

Now taking $(i, j) \in B$, we have,

$$|\hbar(x) - \hbar(y)| \leq |\hbar(x) - x_{ij}| + |x_{ij} - y_{ij}| + |y_{ij} - \hbar(y)| \leq 3\|x - y\|_*.$$

Thus \hbar is a Lipschitz function.

For the space ${}_2m^I_{\mathcal{Z}_0}(F)$ the result can be proved similarly. \square

Theorem 10. *If $x, y \in {}_2m^I_{\mathcal{Z}}(F)$, then $(x.y) \in {}_2m^I_{\mathcal{Z}}(F)$ and $\hbar(xy) = \hbar(x)\hbar(y)$.*

Proof. For $\epsilon > 0$

$$B_x = \{(i, j) \in \mathbb{N} \times \mathbb{N} : |x_{ij} - \hbar(x)| < \epsilon\} \in \mathcal{F}(I),$$

$$B_y = \{(i, j) \in \mathbb{N} \times \mathbb{N} : |y_{ij} - \hbar(y)| < \epsilon\} \in \mathcal{F}(I).$$

Now,

$$|x_{ij}y_{ij} - \hbar(x)\hbar(y)| = |x_{ij}y_{ij} - x_{ij}\hbar(y) + x_{ij}\hbar(y) - \hbar(x)\hbar(y)| \leq |x_{ij}||y_{ij} - \hbar(y)| + |\hbar(y)||x_{ij} - \hbar(x)| \quad (2.9)$$

As ${}_2m^I_{\mathcal{Z}}(F) \subseteq {}_2\mathcal{Z}^I_\infty(F)$, there exists an $M \in \mathbb{R}$ such that $|x_{ij}| < M$ and $|\hbar(y)| < M$.

Using equation (2.9), we get

$$|x_{ij}y_{ij} - \hbar(x)\hbar(y)| \leq M\epsilon + M\epsilon = 2M\epsilon$$

For all $(i, j) \in B_x \cap B_y \in {}_2m^I(F)$.

Hence $(x.y) \in {}_2m^I_{\mathcal{Z}}(F)$ and $\hbar(xy) = \hbar(x)\hbar(y)$.

For the space ${}_2m^I_{\mathcal{Z}_0}(F)$ the result can be proved similarly. \square

Acknowledgments: The authors would like to record their gratitude to the reviewer for his careful reading and making some useful corrections which improved the presentation of the paper.

References

- Altay, B., F. Başar and M. Mursaleen (2006). On the Euler sequence space which include the spaces which include the spaces l_p and l_∞ . *Inform. Sci.* **176**(10), 1450–1462.
- Başar, F. and B. Altay (2003). On the spaces of sequences of p-bounded variation and related matrix mappings. *Ukrainian Math.J.* **55**(1), 136–147.
- Buck, R.C. (1953). Generalized asymptotic density. *Amer.J.Math.* **75**(1), 335–346.
- Connor, J.S. (1989). On strong matrix summability with respect to a modulus and statistical convergence. *Canad.Math.Bull.* **32**, 194–198.
- Connor, J.S. (1998). The statistical and strong P-Cesaro convergence of sequences. *Analysis* **8**, 47–63.
- Connor, J.S. and J. Kline (1996). On statistical limit points and the consistency of statistical convergence. *J.Math.Anal.Appl.* **197**, 392–399.
- Connor, J.S., J.A. Fridy and J. Kline (1994). Statistically Pre-Cauchy sequence. *Analysis* **14**, 311–317.
- Demirci, K. (2001). I-limit superior and limit inferior. *Math. Commun.* (6), 165–172.
- Fast, H. (1951). Sur la convergence statistique. *Colloq. Math.* **2**, 241–244.
- Fridy, J.A. (1985). On statistical convergence. *Analysis* **5**, 301–313.
- Fridy, J.A. (1993). Statistical limit points. *Proc.Amer.Math.Soc.* **11**, 1187–1192.
- Garling, D.J.H. (1966). On symmetric sequence spaces. *Proc.London. Math.Soc.* **16**, 85–106.
- Khan, V.A. and N. Khan (2013). On some I- Convergent double sequence spaces defined by a modulus function. *Engineering,Scientific Research* **5**, 35–40.
- Khan, V.A., Khalid Ebadullah, A. Esi and M. Shafiq (2013). On Zeweir I-convergent sequence spaces defined by a modulus function. *Afrika Matematika* **3**(2), 22–27.
- Kolk, E. (1993). On strong boundedness and summability with respect to a sequence of modulii. *Acta Comment. Univ.Tartu* **960**, 41–50.
- Kolk, E. (1994). Inclusion theorems for some sequence spaces defined by a sequence of modulii. *Acta Comment. Univ.Tartu* **970**, 65–72.
- Kostyrko, P., T.Šalát and W.Wilczyński (2000). I-convergence. *Real Analysis Exchange* **26**(2), 669–686.
- Köthe, G. (1970). *Topological Vector spaces I*. Springer,Berlin.
- Malkowsky, E. (1997). Recent results in the theory of matrix transformation in sequence spaces. *Math.Vesnik* **49**, 187–196.
- Nakano, H. (1953). Concave modulars. *J. Math Soc. Japan* **5**, 29–49.
- Ng, P.N. and P.Y. Lee (1978). Cesaro sequence spaces of non-absolute type. *Pracc.Math.* **20**(2), 429–433.
- Ruckle, W.H. (1967). Symmetric coordinate spaces and symmetric bases. *Canad.J.Math.* **19**, 828–838.
- Ruckle, W.H. (1968). On perfect symmetric BK-spaces. *Math.Ann.* **175**, 121–126.
- Ruckle, W.H. (1973). FK-spaces in which the sequence of coordinate vectors is bounded. *Canad.J.Math.* **25**(5), 873–875.
- Šalát, T., B.C. Tripathy and M. Ziman (2004). On some properties of I-convergence. *Tatra Mt. Math. Publ.* **28**, 279–286.
- Schoenberg, I. J. (1959). The integrability of certain functions and related summability methods. *Amer.Math.Monthly* **66**, 361–375.
- Şengönül, M. (2007). On the zweier sequence space. *Demonstratio Mathematica* (1), 181–196.
- Tripathy, B.C. and B. Hazarika (2011). Some I-Convergent sequence spaces defined by Orlicz function. *Acta Mathematicae Applicatae Sinica* **27**(1), 149–154.
- Wang, C.S. (1978). On nörlund sequence spaces. *Tamkang J.Math.* **9**, 269–274.



Some Families of q -Series Identities and Associated Continued Fractions

H. M. Srivastava^{a,*}, S. N. Singh^b, S. P. Singh^b

^a*Department of Mathematics and Statistics, University of Victoria, Victoria, British Columbia V8W 3R4, Canada
and China Medical University, Taichung 40402, Taiwan, Republic of China*

^b*Department of Mathematics, Tilak Dhari Post-Graduate College, Jaunpur 222002, Uttar Pradesh, India*

Abstract

In this paper, by using some known q -identities, the authors derive several results involving q -series and associated continued fractions. Some other closely-related q -identities are also considered.

Keywords: q -Series, q -Identities, q -Series identities, Rogers-Ramanujan identities, Continued fractions.
2010 MSC: Primary 11A55, 33D90; Secondary 11F20.

1. Introduction, Definitions and Notations

For $q, \lambda, \mu \in \mathbb{C}$ ($|q| < 1$), the basic (or q -) shifted factorial $(\lambda; q)_\mu$ is defined by (see, for example, (Slater, 1966); see also the recent works (Cao & Srivastava, 2013), (Choi & Srivastava, 2014), (Srivastava, 2011), (Srivastava & Choi, 2012) and (Srivastava & Karlsson, 1985) dealing with the q -analysis)

$$(\lambda; q)_\mu := \prod_{j=0}^{\infty} \left(\frac{1 - \lambda q^j}{1 - \lambda q^{\mu+j}} \right) \quad (|q| < 1; \lambda, \mu \in \mathbb{C}), \quad (1.1)$$

so that

$$(\lambda; q)_n := \begin{cases} 1 & (n = 0) \\ \prod_{j=0}^{n-1} (1 - \lambda q^j) & (n \in \mathbb{N}) \end{cases} \quad (1.2)$$

*Corresponding author

Email addresses: harimsri@math.uvic.ca (H. M. Srivastava), snsp39@gmail.com (S. N. Singh), snsp39@yahoo.com (S. P. Singh)

and

$$(\lambda; q)_\infty := \prod_{j=0}^{\infty} (1 - \lambda q^j) \quad (|q| < 1; \lambda \in \mathbb{C}), \tag{1.3}$$

where, as usual, \mathbb{C} denotes the set of complex numbers and \mathbb{N} denotes the set of positive integers (with $\mathbb{N}_0 := \mathbb{N} \cup \{0\}$). For convenience, we write

$$(a_1, \dots, a_k; q)_n = (a_1; q)_n \cdots (a_k; q)_n \tag{1.4}$$

and

$$(a_1, \dots, a_k; q)_\infty = (a_1; q)_\infty \cdots (a_k; q)_\infty. \tag{1.5}$$

In the literature on q -series, there usually are two types of identities as follows:

Type 1. Series = Product

and

Type 2. Series = Series.

The most famous identities of Type 1 are the following Rogers-Ramanujan identities:

$$\sum_{n=0}^{\infty} \frac{q^{n^2}}{(q; q)_n} = \frac{1}{(q; q^5)_\infty (q^4; q^5)_\infty} \tag{1.6}$$

and

$$\sum_{n=0}^{\infty} \frac{q^{n(n+1)}}{(q; q)_n} = \frac{1}{(q^2; q^5)_\infty (q^3; q^5)_\infty}. \tag{1.7}$$

The identities (1.6) and (1.7) have a remarkably fascinating history. They were first proved in 1894 by Rogers (Rogers, 1894), but his paper was completely overlooked. They were rediscovered (without any published proof) by Ramanujan sometime before 1913. These identities were discovered again in 1917 and proved independently by Schur (Schur, 1973).

There are numerous q -identities that are similar to the Rogers-Ramanujan identities (1.6) and (1.7). These include (for example) the q -identities due to Jackson (Jackson, 1928), Rogers (see (Rogers, 1894) and (Rogers, 1917)), Bailey (see (Bailey, 1947) and (Bailey, 1949)), and Slater (Slater, 1952) (see also (McLaughlin & Sills, 2009)). In particular, Slater’s paper (Slater, 1952) contains a list of 130 q -identities of the Rogers-Ramanujan type. On the other hand, in terms of continued fractions, Ramanujan stated for $|q| < 1$ that

$$\frac{\sum_{n=0}^{\infty} \frac{q^{n^2}}{(q; q)_n}}{\sum_{n=0}^{\infty} \frac{q^{n(n+1)}}{(q; q)_n}} = 1 + \frac{q}{1+} \frac{q^2}{1+} \frac{q^3}{1+} \cdots. \tag{1.8}$$

There are numerous q -identities of Type 2 in the ‘Lost’ Notebook of Ramanujan (see (Ramanujan, 1988)) and also in other places in the literature on q -series. Our aim in this paper is to consider various q -identities of Type 2 in order to establish a number of results involving continued fractions of the form involved in (1.8).

2. A Set of Main Results

In this section, we propose to derive continued-fraction expressions for the quotients of the series involved in some known q -identities.

First of all, we consider the following identity (see (Bowman & McLaughlin, 2006, p. 4, Theorem 1, Eq. (2.10)) and (McLaughlin et al., 2008, p. 41, Eq. (6.1.7))):

$$\sum_{n=0}^{\infty} \frac{q^{n(n-1)}(-\gamma)^n}{(\gamma q; q^2)_n (q^2; q^2)_n} = \frac{1}{(\gamma q; q^2)_{\infty}} \sum_{n=0}^{\infty} \frac{q^{n(n-1)}(-\gamma)^n}{(q; q)_n}. \tag{2.1}$$

and its companion identity given by (see (Bowman & McLaughlin, 2006, p. 4, Theorem 1, Eq. (2.11)) and (McLaughlin et al., 2008, p. 41, Eq. (6.1.8))):

$$\sum_{n=0}^{\infty} \frac{q^{n(n-1)}(-\gamma)^n}{(\gamma/q; q^2)_n (q^2; q^2)_n} = \frac{1}{(\gamma/q; q^2)_{\infty}} \sum_{n=0}^{\infty} \frac{q^{n(n-2)}(-\gamma)^n}{(q; q)_n}. \tag{2.2}$$

I. We now investigate the quotient of the right-hand sides of (2.1) and (2.2) as follows:

$$\begin{aligned} \frac{(\gamma/q; q^2)_{\infty}}{(\gamma q; q^2)_{\infty}} \frac{\sum_{n=0}^{\infty} \frac{q^{n(n-1)}(-\gamma)^n}{(q; q)_n}}{\sum_{n=0}^{\infty} \frac{q^{n(n-2)}(-\gamma)^n}{(q; q)_n}} &= \frac{1 - \frac{\gamma}{q}}{\sum_{n=0}^{\infty} \frac{q^{n(n-2)}(-\gamma)^n}{(q; q)_n}} \\ &= \frac{\sum_{n=0}^{\infty} \frac{q^{n(n-1)}(-\gamma)^n}{(q; q)_n}}{\sum_{n=0}^{\infty} \frac{q^{n(n-2)}(-\gamma)^n}{(q; q)_n} - \sum_{n=0}^{\infty} \frac{q^{n(n-1)}(-\gamma)^n}{(q; q)_n}} \\ &= \frac{1 - \frac{\gamma}{q}}{1 + \frac{\sum_{n=0}^{\infty} \frac{q^{n(n-1)}(-\gamma)^n}{(q; q)_n}}{\sum_{n=0}^{\infty} \frac{q^{n(n-2)}(-\gamma)^n}{(q; q)_n}}} \\ &= \frac{1 - \frac{\gamma}{q}}{1 + \frac{\sum_{n=0}^{\infty} \frac{(-\gamma)^n q^{n(n-2)} (1 - q^n)}{(q; q)_n}}{\sum_{n=0}^{\infty} \frac{(-\gamma)^n q^{n(n-1)}}{(q; q)_n}}} \end{aligned}$$

$$\begin{aligned}
 &= \frac{1 - \frac{\gamma}{q}}{1 + \frac{\sum_{n=1}^{\infty} \frac{(-\gamma)^n q^{n(n-2)}}{(q; q)_{n-1}}}{\sum_{n=0}^{\infty} \frac{(-\gamma)^n q^{n(n-1)}}{(q; q)_n}}} \\
 &= \frac{1 - \frac{\gamma}{q}}{1 + \frac{(-\gamma/q)}{\sum_{n=0}^{\infty} \frac{(-\gamma)^n q^{n(n-1)}}{(q; q)_n}}}. \tag{2.3} \\
 &\qquad\qquad\qquad \frac{\sum_{n=0}^{\infty} \frac{(-\gamma)^n q^{n^2}}{(q; q)_n}}
 \end{aligned}$$

Proceeding in the same way, we find that

$$\frac{(\gamma/q; q^2)_{\infty} \sum_{n=0}^{\infty} \frac{q^{n(n-1)}(-\gamma)^n}{(q; q)_n}}{(\gamma q; q^2)_{\infty} \sum_{n=0}^{\infty} \frac{q^{n(n-2)}(-\gamma)^n}{(q; q)_n}} = \frac{1 - \frac{\gamma}{q}}{1 - \frac{\gamma}{q}} \frac{\gamma}{1 - \frac{\gamma}{q}} \frac{\gamma q}{1 - \frac{\gamma}{q}} \frac{\gamma q^2}{1 - \frac{\gamma}{q}} \frac{\gamma q^3}{1 - \frac{\gamma}{q}} \dots \tag{2.4}$$

From (2.1), (2.2) and (2.4), we have

$$\frac{\sum_{n=0}^{\infty} \frac{q^{n(n-1)}(-\gamma)^n}{(\gamma q; q^2)_n (q^2; q^2)_n}}{\sum_{n=0}^{\infty} \frac{q^{n(n-1)}(-\gamma)^n}{(\gamma/q; q^2)_n (q^2; q^2)_n}} = \frac{1 - \frac{\gamma}{q}}{1 - \frac{\gamma}{q}} \frac{\gamma}{1 - \frac{\gamma}{q}} \frac{\gamma q}{1 - \frac{\gamma}{q}} \frac{\gamma q^2}{1 - \frac{\gamma}{q}} \frac{\gamma q^3}{1 - \frac{\gamma}{q}} \dots \tag{2.5}$$

The following special cases and consequences of (2.5) are worthy of note. Firstly, upon setting $\gamma = -q$ in (2.5), we get

$$\frac{\sum_{n=0}^{\infty} \frac{q^{n^2}}{(q^4; q^4)_n}}{\sum_{n=0}^{\infty} \frac{q^{n^2}}{(-1; q^2)_n (q^2; q^2)_n}} = \frac{2}{1 + \frac{1}{1 + \frac{q}{1 + \frac{q^2}{1 + \dots}}}} \tag{2.6}$$

which, in light of the known result (Andrews & Berndt, 2005, p. 87, Entry (3.2.3)), yields

$$\frac{2(q; q^5)_{\infty} (q^4; q^5)_{\infty}}{(q^2; q^4)_{\infty}} \sum_{n=0}^{\infty} \frac{q^{n^2}}{(-1; q^2)_n (q^2; q^2)_n} = 1 + \frac{1}{1 + \frac{q}{1 + \frac{q^2}{1 + \dots}}} \tag{2.7}$$

If we use another known result (Andrews & Berndt, 2005, p. 153, Corollary (6.2.6)) in (2.7), we obtain

$$2 \sum_{n=0}^{\infty} \frac{q^{n^2}}{(-1; q^2)_n (q^2; q^2)_n} = \frac{(q^2; q^4)_{\infty}}{(q; q^5)_{\infty} (q^4; q^5)_{\infty}} + \frac{(q^2; q^4)_{\infty}}{(q^2; q^5)_{\infty} (q^3; q^5)_{\infty}}. \tag{2.8}$$

We next set $\gamma = -q^3$ in (2.5) and obtain

$$\frac{\sum_{n=0}^{\infty} \frac{q^{n^2+2n}}{(-q^4; q^2)_n (q^2; q^2)_n}}{\sum_{n=0}^{\infty} \frac{q^{n(n+2)}}{(q^4; q^4)_n}} = \frac{1 + q^2}{1 +} \frac{q^2}{1 +} \frac{q^3}{1 +} \frac{q^4}{1 +} \dots, \tag{2.9}$$

which, in conjunction with a known result (Andrews & Berndt, 2005, p. 87, Entry (3.2.3)), yields the following consequence of (2.5):

$$\frac{(q^2; q^5)_{\infty} (q^3; q^5)_{\infty}}{(q^2; q^4)_{\infty}} \sum_{n=0}^{\infty} \frac{q^{n(n+2)}}{(-q^2; q^2)_{n+1} (q^2; q^2)_n} = \frac{1}{1 +} \frac{q^2}{1 +} \frac{q^3}{1 +} \frac{q^4}{1 +} \dots. \tag{2.10}$$

II. Let us consider the following q -identity of Type 2 (see (Bowman & McLaughlin, 2006, p. 4, Theorem 1, Eq. (2.7)) and (McLaughlin et al., 2008, p. 40, Eq. (6.1.4))):

$$\sum_{n=0}^{\infty} \frac{(a; q)_n}{(q; q)_n} \gamma^n q^{n(n-1)/2} = (-\gamma; q)_{\infty} \sum_{n=0}^{\infty} \frac{(-a\gamma)^n q^{n(n-1)}}{(-\gamma; q)_n (q; q)_n}, \tag{2.11}$$

which, upon replacing γ by γq , yields

$$\sum_{n=0}^{\infty} \frac{(a; q)_n}{(q; q)_n} \gamma^n q^{n(n+1)/2} = (-\gamma q; q)_{\infty} \sum_{n=0}^{\infty} \frac{(-a\gamma)^n q^{n^2}}{(-\gamma q; q)_n (q; q)_n}. \tag{2.12}$$

By taking the quotient of the left-hand sides of (2.11) and (2.12), we find that

$$\begin{aligned} \frac{\sum_{n=0}^{\infty} \frac{(a; q)_n}{(q; q)_n} \gamma^n q^{n(n+1)/2}}{\sum_{n=0}^{\infty} \frac{(a; q)_n}{(q; q)_n} \gamma^n q^{n(n-1)/2}} &= \frac{1}{1 + \frac{\sum_{n=0}^{\infty} \frac{(a; q)_n}{(q; q)_n} \gamma^n q^{n(n-1)/2} - \sum_{n=0}^{\infty} \frac{(a; q)_n}{(q; q)_n} \gamma^n q^{n(n+1)/2}}{\sum_{n=0}^{\infty} \frac{(a; q)_n}{(q; q)_n} \gamma^n q^{n(n+1)/2}}} \\ &= \frac{1}{1 + \frac{\sum_{n=1}^{\infty} \frac{(a; q)_n}{(q; q)_{n-1}} \gamma^n q^{n(n-1)/2}}{\sum_{n=0}^{\infty} \frac{(a; q)_n}{(q; q)_n} \gamma^n q^{n(n+1)/2}}} \end{aligned}$$

$$\cdot \frac{1}{1 + \frac{\gamma^{(1-a)}}{\sum_{n=0}^{\infty} \frac{(a; q)_n}{(q; q)_n} \gamma^n q^{n(n+1)/2}}}. \tag{2.13}$$

$$\frac{\sum_{n=0}^{\infty} \frac{(aq; q)_n}{(q; q)_n} \gamma^n q^{n(n+1)/2}}{\sum_{n=0}^{\infty} \frac{(aq; q)_n}{(q; q)_n} \gamma^n q^{n(n+1)/2}}$$

It is easily observed that

$$\frac{\sum_{n=0}^{\infty} \frac{(a; q)_n}{(q; q)_n} \gamma^n q^{n(n+1)/2}}{\sum_{n=0}^{\infty} \frac{(aq; q)_n}{(q; q)_n} \gamma^n q^{n(n+1)/2}} = 1 + \frac{\sum_{n=0}^{\infty} \frac{(a; q)_n}{(q; q)_n} \gamma^n q^{n(n+1)/2} - \sum_{n=0}^{\infty} \frac{(aq; q)_n}{(q; q)_n} \gamma^n q^{n(n+1)/2}}{\sum_{n=0}^{\infty} \frac{(aq; q)_n}{(q; q)_n} \gamma^n q^{n(n+1)/2}}$$

$$= 1 + \frac{\sum_{n=0}^{\infty} \frac{(aq; q)_{n-1}(-a)(1 - q^n)}{(q; q)_n} \gamma^n q^{n(n+1)/2}}{\sum_{n=0}^{\infty} \frac{(aq; q)_n}{(q; q)_n} \gamma^n q^{n(n+1)/2}}$$

$$= 1 - \frac{a\gamma q}{\frac{\sum_{n=0}^{\infty} \frac{(aq; q)_n}{(q; q)_n} \gamma^n q^{n(n+1)/2}}{\sum_{n=0}^{\infty} \frac{(aq; q)_n}{(q; q)_n} \gamma^n q^{n(n+3)/2}}}, \tag{2.14}$$

which, when combined with (2.14), yields

$$\frac{\sum_{n=0}^{\infty} \frac{(a; q)_n}{(q; q)_n} \gamma^n q^{n(n+1)/2}}{\sum_{n=0}^{\infty} \frac{(a; q)_n}{(q; q)_n} \gamma^n q^{n(n-1)/2}} = \frac{1}{1+} \frac{\gamma(1-a)}{1-} \frac{a\gamma q}{\frac{\sum_{n=0}^{\infty} \frac{(aq; q)_n}{(q; q)_n} \gamma^n q^{n(n+1)/2}}{\sum_{n=0}^{\infty} \frac{(aq; q)_n}{(q; q)_n} \gamma^n q^{n(n+3)/2}}}. \tag{2.15}$$

Finally, by iterating the above process, we get the following result:

$$\frac{\sum_{n=0}^{\infty} \frac{(a; q)_n}{(q; q)_n} \gamma^n q^{n(n+1)/2}}{\sum_{n=0}^{\infty} \frac{(a; q)_n}{(q; q)_n} \gamma^n q^{n(n-1)/2}} = \frac{1}{1+} \frac{\gamma(1-a)}{1-} \frac{a\gamma q}{1+} \frac{\gamma q(1-aq)}{1-} \frac{a\gamma q^3}{1+} \dots \tag{2.16}$$

Applying the q -identities (2.11), (2.12) and (2.16), we find that

$$\frac{\sum_{n=0}^{\infty} \frac{(-a\gamma)^n q^{n^2}}{(-\gamma q; q)_n (q; q)_n}}{\sum_{n=0}^{\infty} \frac{(-a\gamma)^n q^{n(n-1)}}{(-\gamma; q)_n (q; q)_n}} = \frac{(1+\gamma)}{1+} \frac{\gamma(1-a)}{1-} \frac{a\gamma q}{1+} \frac{\gamma q(1-aq)}{1-} \frac{a\gamma q^3}{1+} \dots, \tag{2.17}$$

which, upon setting $\gamma = -q$, yields

$$\frac{\sum_{n=0}^{\infty} \frac{a^n q^{n(n+1)}}{(q^2; q)_n (q; q)_n}}{\sum_{n=0}^{\infty} \frac{a^n q^{n^2}}{[(q; q)_n]^2}} = \frac{(1-q)}{1-} \frac{q(1-a)}{1+} \frac{aq^2}{1-} \frac{q^2(1-aq)}{1+} \frac{aq^4}{1-} \dots. \tag{2.18}$$

In its further special case when $a = 1$, (2.18) yields

$$\sum_{n=0}^{\infty} \frac{q^{n(n+1)}}{(q; q)_{n+1} (q; q)_n} = \sum_{n=0}^{\infty} \frac{q^{n^2}}{(q; q)_n^2} = \frac{1}{(q; q)_{\infty}}. \tag{2.19}$$

For $\gamma = 1$ and $a = -q$, we find from (2.17) that

$$\frac{\sum_{n=0}^{\infty} \frac{q^{n(n+1)}}{(q^2; q^2)_n}}{\sum_{n=0}^{\infty} \frac{q^{n^2}}{(-1; q)_n (q; q)_n}} = \frac{2}{1+} \frac{(1+q)}{1+} \frac{q^2}{1+} \frac{q(1+q^2)}{1+} \frac{q^4}{1+} \dots. \tag{2.20}$$

If, instead, we put $\gamma = 1$ and $a = q$ in (2.17), we get

$$\frac{\sum_{n=0}^{\infty} \frac{(-1)^n q^{n(n+1)}}{(q^2; q^2)_n}}{\sum_{n=0}^{\infty} \frac{(-1)^n q^{n^2}}{(-1; q)_n (q; q)_n}} = \frac{2}{1+} \frac{(1-q)}{1-} \frac{q^2}{1+} \frac{q(1-q^2)}{1-} \frac{q^4}{1+} \dots. \tag{2.21}$$

We now recall a known result (Andrews & Berndt, 2005, p. 152, Entry (6.2.32)) with $a = -1$ as follows:

$$\sum_{n=0}^{\infty} \frac{(-1)^n q^{n(n+1)}}{(q^2; q^2)_n} = \frac{1}{(-q; q)_{\infty}} \sum_{n=0}^{\infty} q^{n(n+1)/2} = (q^2; q^2)_{\infty}, \tag{2.22}$$

which, in combination with (2.21), yields

$$\frac{2}{(q^2; q^2)_{\infty}} \sum_{n=0}^{\infty} \frac{(-1)^n q^{n^2}}{(-1; q)_n (q; q)_n} = 1 + \frac{1-q}{1-} \frac{q^2}{1+} \frac{q(1-q^2)}{1-} \frac{q^4}{1+} \dots. \tag{2.23}$$

III. Let us consider the following known q -identity (see (Bowman & McLaughlin, 2006, p. 4, Theorem 1, Eq. (2.9)) and (McLaughlin et al., 2008, p. 40, Eq. (6.1.6))):

$$\sum_{n=0}^{\infty} \frac{q^{3n(n-1)/2} \gamma^n}{(\gamma; q^2)_n (q; q)_n} = \frac{1}{(\gamma; q^2)_{\infty}} \sum_{n=0}^{\infty} \frac{q^{n(2n-1)} \gamma^n}{(q^2; q^2)_n}, \tag{2.24}$$

which, upon replacing γ by γq^2 , yields

$$\sum_{n=0}^{\infty} \frac{q^{n(3n+1)/2} \gamma^n}{(\gamma q^2; q^2)_n (q; q)_n} = \frac{1}{(\gamma q^2; q^2)_{\infty}} \sum_{n=0}^{\infty} \frac{q^{n(2n+1)} \gamma^n}{(q^2; q^2)_n}. \tag{2.25}$$

For the quotient of of the right-hand sides of (2.24) and (2.25), we have

$$\begin{aligned} \frac{(1 - \gamma) \sum_{n=0}^{\infty} \frac{q^{n(2n+1)} \gamma^n}{(q^2; q^2)_n}}{\sum_{n=0}^{\infty} \frac{q^{2n^2-n} \gamma^n}{(q^2; q^2)_n}} &= \frac{(1 - \gamma)}{1 + \frac{\sum_{n=0}^{\infty} \frac{\gamma^n q^{n(2n-1)}}{(q^2; q^2)_n} (1 - q^{2n})}{\sum_{n=0}^{\infty} \frac{\gamma^n q^{n(2n+1)}}{(q^2; q^2)_n}}} \\ &= \frac{(1 - \gamma)}{1 + \frac{\sum_{n=1}^{\infty} \frac{\gamma^n q^{n(2n-1)}}{(q^2; q^2)_{n-1}}}{\sum_{n=0}^{\infty} \frac{\gamma^n q^{n(2n-1)}}{(q^2; q^2)_n}}} = \frac{(1 - \gamma)}{1 + \frac{\frac{\gamma q}{\sum_{n=0}^{\infty} \frac{\gamma^n q^{n(2n+1)}}{(q^2; q^2)_n}}}{\sum_{n=0}^{\infty} \frac{\gamma^n q^{n(2n+3)}}{(q^2; q^2)_n}}}. \end{aligned} \tag{2.26}$$

Proceeding in the above way, we obtain

$$\frac{(1 - \gamma) \sum_{n=0}^{\infty} \frac{q^{n(2n+1)} \gamma^n}{(q^2; q^2)_n}}{\sum_{n=0}^{\infty} \frac{q^{n(2n-1)} \gamma^n}{(q^2; q^2)_n}} = \frac{(1 - \gamma)}{1 +} \frac{\gamma q}{1 +} \frac{\gamma q^3}{1 +} \frac{\gamma q^5}{1 +} \frac{\gamma q^7}{1 +} \dots \tag{2.27}$$

Finally, by applying (2.24), (2.25) and (2.27), we get

$$\frac{\sum_{n=0}^{\infty} \frac{q^{n(3n+1)/2} \gamma^n}{(\gamma q^2; q^2)_n (q^2; q^2)_n}}{\sum_{n=0}^{\infty} \frac{q^{3n(n-1)/2} \gamma^n}{(\gamma; q^2)_n (q^2; q^2)_n}} = \frac{1 - \gamma}{1 +} \frac{\gamma q}{1 +} \frac{\gamma q^3}{1 +} \frac{\gamma q^5}{1 +} \frac{\gamma q^7}{1 +} \dots \tag{2.28}$$

In its special case when $\gamma = -1$, we find from (2.28) that

$$\frac{\sum_{n=0}^{\infty} \frac{(-1)^n q^{n(3n+1)/2}}{(q^4; q^4)_n}}{\sum_{n=0}^{\infty} \frac{(-1)^n q^{3n(n-1)/2}}{(-1; q^2)_n (q^2; q^2)_n}} = \frac{2}{1-} \frac{q}{1-} \frac{q^3}{1-} \frac{q^5}{1-} \frac{q^7}{1-} \dots \quad (2.29)$$

Many other similar results involving q -series and associated continued fractions can also be derived analogously.

3. Concluding Remarks and Observations

While q -identities of Type 1 include such important and widely-investigated results as the celebrated Rogers-Ramanujan identities, we have successfully derived several families of q -identities of Type 2 involving q -series and associated continued fractions. We have also considered some other closely-related q -identities of Types 1 and 2.

Such q -series identities of Type 2 as (for example) (2.1), (2.2), (2.11) and (2.24), upon which our present investigation depends remarkably heavily, are derivable as special or limit cases of relatively more familiar known q -identities (see, for details, (Bowman & McLaughlin, 2006, pp. 4–7) and (McLaughlin et al., 2008, p. 42)).

Acknowledgements

The second-named author (S. N. Singh) is thankful to The Department of Science and Technology of the Government of India (New Delhi, India) for support under a major research project No. SR/S4/MS:735/2011 dated 07 May 2013, entitled “A Study of Transformation Theory of q -Series, Modular Equations, Continued Fractions and Ramanujan’s Mock-Theta Functions,” under which this work was done.

References

- Andrews, G. E. and B. C. Berndt (2005). *Ramanujan’s Lost Notebook, Part I*. Springer, Berlin, Heidelberg and New York.
- Bailey, W. N. (1947). Some identities in combinatory analysis. *Proc. London Math. Soc. (Ser. 2)* **49**, 421–435.
- Bailey, W. N. (1949). Identities of the Rogers-Ramanujan type. *Proc. London Math. Soc. (Ser. 2)* **50**, 1–10.
- Bowman, D. and J. McLaughlin (2006). Some more identities of the Rogers-Ramanujan type. Preprint 2006 [arXiv:math/0607202v2 [math.NT] 8 Jul 2006].
- Cao, J. and H. M. Srivastava (2013). Some q -generating functions of the Carlitz and Srivastava-Agarwal types associated with the generalized Hahn polynomials and the generalized Rogers-Szegő polynomials. *Appl. Math. Comput.* **219**, 8398–8406.
- Choi, J. and H. M. Srivastava (2014). q -extensions of a multivariable and multiparameter generalization of the Gottlieb polynomials in several variables. *Tokyo J. Math.* **37**, 111–125.

- Jackson, F. H. (1928). Examples of a generalization of Euler's transformation for power series. *Messenger Math.* **57**, 169–187.
- McLaughlin, J., A. V. Sills and P. Zimmer (2008). Rogers-Ramanujan-Slater type identities. *Electron. J. Combin.* **15**, 1–59. Article ID DS15.
- McLaughlin, J. and A. V. Sills (2009). Some more identities of the Rogers-Ramanujan type. *Ramanujan J.* **18**, 307–325.
- Ramanujan, S. (1988). *The 'Lost' Notebook and Other Unpublished Papers (With an Introduction by G. E. Andrews)*. Springer-Verlag, Berlin, Heidelberg and New York; Narosa Publishing House, New Delhi.
- Rogers, L. J. (1894). Second memoir on the expansion of certain infinite products. *Proc. London Math. Soc. (Ser. 1)* **25**, 318–343.
- Rogers, L. J. (1917). On two theorems of combinatory analysis and some allied identities. *Proc. London Math. Soc. (Ser. 2)* **16**, 315–336.
- Schur, I. (1973). Ein Beitrag zur additiven Zahlentheorie und zur Theorie der Kettenbrüche. In: *Gesammelte Abhandlungen, Band II*. Springer-Verlag, Berlin, Heidelberg and New York. pp. 117–136. [Originally published in *Sitzungsberichte der Preussischen Akademie der Wissenschaften: Physikalisch-Mathematische Klasse* (1917), 302–321].
- Slater, L. J. (1952). Further identities of the Rogers-Ramanujan type. *Proc. London Math. Soc. (Ser. 2)* **54**, 147–167.
- Slater, L. J. (1966). *Generalized Hypergeometric Functions*. Cambridge University Press, Cambridge, London and New York.
- Srivastava, H. M. (2011). Some generalizations and basic (or q -) extensions of the Bernoulli, Euler and Genocchi polynomials. *Appl. Math. Inform. Sci.* **5**, 390–444.
- Srivastava, H. M. and J. Choi (2012). *Zeta and q -Zeta Functions and Associated Series and Integrals*. Elsevier Science Publishers, Amsterdam, London and New York.
- Srivastava, H. M. and P. W. Karlsson (1985). *Multiple Gaussian Hypergeometric Series*. Halsted Press (Ellis Horwood Limited, Chichester), John Wiley and Sons, New York, Chichester, Brisbane and Toronto.



Primality Testing and Factorization by using Fourier Spectrum of the Riemann Zeta Function

Takaaki Musha^{a,*}

^a *Advanced Science-Technology Research Organization 3-11-7-601,
Namiki, Kanazawa-ku, Yokohama 236-0005 Japan*

Abstract

In number theory, integer factorization is the decomposition of a composite number into a product of smaller integers, for which there is not known efficient algorithm. In this article, the author tries to make primality testing and factorization of integers by using Fourier transform of a correlation function generated from the Riemann zeta function. From the theoretical analysis, we can see that prime factorization for the integer composed of two different primes can be conducted within a polynomial time and it can be seen that this special case belongs to the P class.

Keywords: Primality testing, prime factorization, Fourier transform, Riemann zeta function.
2010 MSC: 11A51, 11M06, 11Y05, 11Y11, 42A38.

1. Introduction

In number theory, integer factorization is the decomposition of a composite number into a product of smaller integers. When the numbers are very large, no efficient, non-quantum integer factorization algorithm is known. However, it has not been proven that no efficient algorithm exists (Klee & Wagon, 1991). The presumed difficulty of this problem is at the heart of widely used algorithms in cryptography such as RSA. Many areas of mathematics and computer science have been brought to bear on the problem, including elliptic curves, algebraic number theory, and quantum computing.

Recently, Shor's algorithm has been proposed by Peter Shor, which is a quantum algorithm for integer factorization. On a quantum computer, it has been proved that Shor's algorithm runs in polynomial time (Shor, 1997). But the polynomial time factoring algorithm of integers has not been found for ordinary computing systems.

*Corresponding author

Email address: takaaki.mushya@gmail.com (Takaaki Musha)

In optics, we know that white light consists of all visible frequencies mixed together and the prism breaks them apart so we can see the separate frequencies. It is like the Riemann zeta function be consisted of primes shown as

$$\zeta(s) = \sum_{n=1}^{\infty} \frac{1}{n^s} = \prod_p \frac{1}{(1 - p^{-s})}$$

where p runs over all of primes.

From which, we can consider the white light as a zeta function and separate component frequencies are primes. As we use a prism to decompose visible light into components of different frequencies, we can use Fourier transforms as a prism to decompose the zeta function into primes. In this paper, the method of primality testing and prime factorization by using Fourier transforms of the Riemann zeta function is presented.

2. Frequency spectrum of a correlation function generated from the Riemann Zeta function

Riemann zeta function is an analytic function defined by $\zeta(s) = \sum_{n=1}^{\infty} n^{-s}$. From which, we define the Fourier transform of $z_{\sigma}(t, \tau)$ shown as

$$Z_{\sigma}(t, \omega) = \lim_{T \rightarrow \infty} \int_{-T}^{+T} z_{\sigma}(t, \tau) e^{-i\omega\tau} d\tau, \tag{2.1}$$

where $z_{\sigma}(t, \tau)$ is a time-dependent autocorrelation function (Yen, 1987) given by

$$z_{\sigma}(t, \tau) = \zeta(\sigma - i(t + \tau/2)) \cdot \zeta^*(\sigma - i(t - \tau/2)). \tag{2.2}$$

In this formula, $\zeta^*(s)$ is a conjugate of $\zeta(s)$.

In the previous paper of author's (Musha, 2014), $Z_{\sigma}(t, \omega)$ can be shown as

$$Z_{\sigma}(t, \omega) = \sum_{n=1}^{\infty} \frac{a(n, t)}{n^{\sigma}} 2\pi\delta(\omega - \frac{1}{2} \log n), \tag{2.3}$$

where $a(n, t)$ is a real valued function given by $a(n, t) = \sum_{n=kl} \cos[\log(k/l)t]$ and $\delta(\omega)$ is a Dirac's delta function.

As $a(n, t)$ is a multiplicative on n which satisfies $a(mn, t) = a(m, t)a(n, t)$ for the case when satisfying $(m, n) = 1$, we have the following equation for the integer n given by $n = p^a q^b r^c \dots$ (Musha, 2012):

$$Z_{\sigma}\left(t, \frac{1}{2} \log n\right) = \frac{2\pi\delta(0)}{n^{\sigma}} \frac{\sin[(a + 1)t \log p]}{\sin(t \log p)} \cdot \frac{\sin[(b + 1)t \log q]}{\sin(t \log q)} \frac{\sin[(c + 1)t \log r]}{\sin(t \log r)} \dots \tag{2.4}$$

From the Fourier transform of $Z_{\sigma}\left(t, \frac{1}{2} \log n\right)$ given by $F_n(\omega) = \int_{-\infty}^{+\infty} Z_{\sigma}\left(t, \frac{1}{2} \log n\right) e^{-i\omega t} dt$, we can obtain the following Lemma.

Lemma 1. If $n = p_1 p_2 p_3 \cdots p_k$, where $p_1, p_2, p_3, \dots, p_k$ are different primes, $F_n(\omega)$ for $\omega > 0$ is consisted of 2^{k-1} discrete spectrum shown as:

$$F_n(\omega) = 2\pi \sum_{i=1}^{2^k} \delta(\omega - \lambda_{i_1} \log p_1 - \lambda_{i_2} \log p_2 - \cdots - \lambda_{i_k} \log p_k), \tag{2.5}$$

where λ_{i_k} equals to -1 or $+1$.

Proof.

As $a(n, t) = \sum_{i=1}^{2^k} [\cos(\lambda_{i_1} \log p_1) + \cos(\lambda_{i_2} \log p_2) + \cdots + \cos(\lambda_{i_k} \log p_k)]$, where $\log p_1, \log p_2, \log p_3, \dots, \log p_k$ are linearly independent over \mathbb{Z} (Kac, 1959), thus $F_n(\omega)$ is consisted of 2^{k-1} different spectrum shown as

$$F_n(\omega) = 2\pi \sum_{i=1}^{2^k} \delta(\omega - \lambda_{i_1} \log p_1 - \lambda_{i_2} \log p_2 - \cdots - \lambda_{i_k} \log p_k) + 2\pi \sum_{i=1}^{2^k} \delta(\omega + \lambda_{i_1} \log p_1 + \lambda_{i_2} \log p_2 + \cdots + \lambda_{i_k} \log p_k).$$

□

Then we obtain following Theorems.

Theorem 1. If and only $F_n(\omega)$ is consisted of a single spectra for $\omega \geq 0$, then n is a prime.

Proof.

It is clear from Lemma 1.

□

Theorem 2. If and only $F_n(\omega)$ is consisted of two spectrum for $\omega \geq 0$, then n has either form of $n = p \cdot q$ ($p \neq q$), $n = p^2$ or $n = p^3$.

Proof.

From Theorem 1, there is only a case for the integer $n = p_1 p_2 \cdots p_k$, when $F_n(\omega)$ is consisted of two spectrum, that is $n = p \cdot q$ ($p \neq q$).

For $r \geq 1$ of the function $a(p^r, t)$:

$$\begin{aligned} r = 1, a(p, t) &= 2 \cos(t \log p) \\ r = 2, a(p^2, t) &= 1 + 2 \cos(2t \log p) \\ r = 3, a(p^3, t) &= 2 \cos(t \log p) + 2 \cos(3t \log p) \\ r = 4, a(p^4, t) &= 1 + 2 \cos(2t \log p) + 2 \cos(4t \log p) \\ r = 5, a(p^5, t) &= 2 \cos(t \log p) + 2 \cos(3t \log p) + 2 \cos(5t \log p) \\ r = 6, a(p^6, t) &= 1 + 2 \cos(2t \log p) + 2 \cos(4t \log p) + 2 \cos(6t \log p) \end{aligned}$$

$$\begin{aligned}
 r = 7, a(p^7, t) &= 2 \cos(t \log p) + 2 \cos(3t \log p) \\
 &\quad + 2 \cos(5t \log p) + 2 \cos(7t \log p) \\
 &\quad \vdots
 \end{aligned}$$

Including the spectra at $\omega = 0$, there are cases for $r = 2$ and $r = 3$ when $a(n, t)$ has two spectrum. □

3. Method for primality testing and factorization by using Fourier spectrum

From Theorems 1 and 2, we can make a primality testing and a factorization of the integer n consisted of two primes from the Fourier spectrum $F_n(\omega)$ ($\omega \geq 0$) by following calculations:

$$\textcircled{1} \quad z_\sigma(t, \tau) = \zeta(\sigma - i(t + \tau/2)) \cdot \zeta^*(\sigma - i(t - \tau/2)), \tag{3.1}$$

$$\textcircled{2} \quad Z_\sigma(t, \omega) = \int_{-\infty}^{+\infty} z_\sigma(t, \tau) e^{-i\omega\tau} d\tau, \tag{3.2}$$

$$\textcircled{3} \quad F_n(\omega) = \int_{-\infty}^{+\infty} Z_\sigma(t, \frac{1}{2} \log n) e^{-i\omega t} dt. \tag{3.3}$$

From which we can obtain the Fourier spectrum by $F_n(\omega) = \int_{-\infty}^{+\infty} Z_\sigma(t, \frac{1}{2} \log n) e^{-i\omega t} dt$. Then we can make a primality testing and integer factorization for an integer n , the process of which is shown in Figure 1.

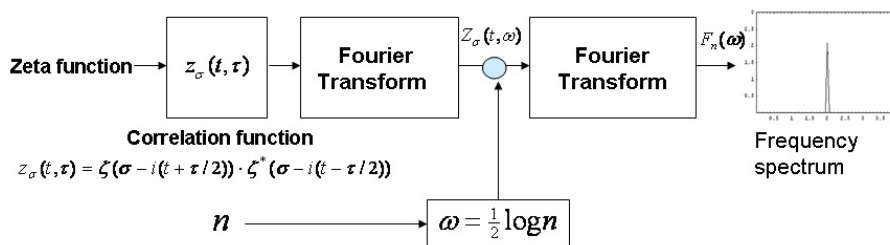


Figure 1. Process to conduct a primality testing for the integer n .

From this process, we can recognize the prime as a single spectra from the frequency analysis result. If there are two spectrum observed from the calculation result, n has either form of $n = p \cdot q$ ($p \neq q$), $n = p^2$ or $n = p^3$ from Theorem 2.

3.1. Numerical Calculation Method to obtain $F_n(\omega)$

To conduct calculations to obtain the values of $F_n(\omega)$ by using discrete Fourier transform, we can select the value for frequency resolution as $\Delta f = 1/4\pi n$ from $\Delta\omega = \left| \frac{1}{2} \log n - \frac{1}{2} \log(n \pm 1) \right| \approx 1/2n$, for large numbers.

Then we select the maximum frequency of DFT analysis to be $f_{\max} = 4 \lceil \log n / 4\pi \rceil$ (where $\lceil \cdot \rceil$ is a Gauss’s symbol), which makes $\omega = \log n$ to be at the center of frequency range.

The element number N for DFT calculation satisfies $f_{\max} = N \cdot \Delta f/2$, then we have $N = [8n \log n] + 1$.

As Eq.(3.1) can be written in a discrete form as

$$z_{\sigma}(m, l) = \zeta(\sigma - i(m\Delta t + l\Delta\tau/2)) \cdot \zeta^*(\sigma - i(m\Delta t - l\Delta\tau/2)). \tag{3.4}$$

From the relation of $\Delta f \cdot \Delta t = 1/N$, we obtain

$$z_{\sigma}(m, l) = \zeta\left(\sigma - i\left(\frac{4\pi n}{N}m + \frac{2\pi n}{N}l\right)\right) \zeta^*\left(\sigma - i\left(\frac{4\pi n}{N}m - \frac{2\pi n}{N}l\right)\right). \tag{3.5}$$

As the total time $T_0 = N \cdot \Delta t = 4\pi n$, then Eq.(3.2) in a discrete form can be given by

$$Z_{\sigma}(m, k) = \frac{4\pi n}{N} \sum_{l=0}^{N-1} z_{\sigma}(m, l) \exp[-i2\pi(k\Delta f) \cdot (l\Delta\tau)]. \tag{3.6}$$

At the frequency of $\omega = \frac{1}{2} \log n$, we have

$$k\Delta f \cdot l\Delta\tau = \frac{\log n}{4\pi} \times \frac{\pi}{2 \log n} l = \frac{l}{8}, \tag{3.7}$$

then we have

$$y(m) = \frac{4\pi n}{N} \sum_{l=0}^{N-1} z_{\sigma}(m, l) \exp\left(-i\frac{\pi}{4}l\right), \tag{3.8}$$

which corresponds to $Z_{\sigma}\left(t, \frac{1}{2} \log n\right)$.

From which, we have the discrete form of Eq.(3.3) given by

$$Y(k) = \frac{4\pi n}{N} \sum_{m=0}^{N-1} y(m) \exp\left(-i2\pi\frac{km}{N}\right), \tag{3.9}$$

which shows the spectrum of $F_n(\omega)$.

Thus we need the following three steps for calculations to obtain $F_n(\omega)$.

- ① Input the integer n and we let $N = [8n \log n] + 1$,
- ② $z_{\sigma}(m, l) = \zeta\left(\sigma - i\left(\frac{4\pi n}{N}m + \frac{2\pi n}{N}l\right)\right) \cdot \zeta^*\left(\sigma - i\left(\frac{4\pi n}{N}m - \frac{2\pi n}{N}l\right)\right)$,
- ③ For $m = 0 \sim N - 1$, calculate $y(m) = \frac{4\pi n}{N} \sum_{l=0}^{N-1} z_{\sigma}(m, l) \exp\left(-i\frac{\pi}{4}l\right)$,
- ④ For $k = 0 \sim N - 1$, calculate $Y(k) = \frac{4\pi n}{N} \sum_{m=0}^{N-1} y(m) \exp\left(-i2\pi\frac{km}{N}\right)$.

3.2. Some examples of primality testing

To confirm the validity of discrete computational algorithm given in this paper, we try to compute some examples shown as follows:

To generate the Riemann zeta function, we use Mathematica by Wolfram Research.

At the calculation, we set $\sigma = 1.1$ to compute $F_n(\omega)$ to minimize the noise generated by DFT calculations.

(Calculation program by using Mathematica).

```

σ=1.1; (Real Part of Zeta function)
n0=17; (Input an Integer)
n1=Ceiling[8*n0*Log[n0]]; (Element number for calculation)
x[m_,l_]:=Zeta[σ-I*(4*Pi*n0*m/n1+2*Pi*n0*l/n1)]*
Conjugate[Zeta[σ-I*(4*Pi*n0*m/n1-2*Pi*n0*l/n1)]]; (Autocorrelation function of
zeta function)
data=Table[N[4*Pi*n0/n1*Sum[x[m,l]*Exp[-I*Pi*l/4],{l,0,n1-1}],{m,0,n1-1}];
ListPlot[Abs[InverseFourier[data]],PlotJoined→True,PlotRange→{{0,n1/2}, {0,200}}, Frame→True]; (DFT calculation and plot results)
    
```

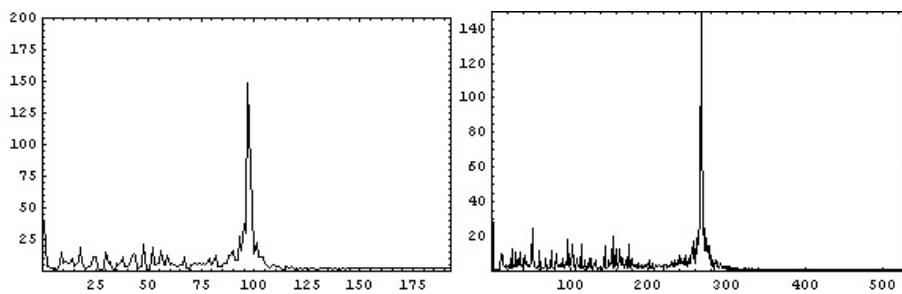


Figure 2. Computational result for $n=17$ (left) and $n=37$ (right).

From calculation, we can see that there exists only one spectrum at the center and it can be shown that both of numbers, 17 and 37 are primes.

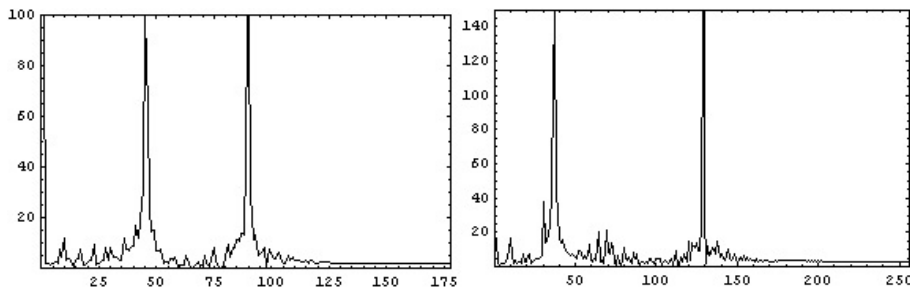


Figure 3. Computational result for $n=16$ (left) and $n=21$ (right).

These results corresponds to $a(p^4, t) = 1 + 2 \cos(2t \log p) + 2 \cos(4t \log p)$ (three spectrum including $\omega = 0$) and $a(p \cdot q, t) = \cos[(\log q - \log p)t] + \cos[(\log q + \log p)t]$ (two spectrum), and we can see that they are composite numbers.

4. Running time for prime factorization by using DFT algorithms

As the value of Riemann zeta function can be generated by the formula (Gourdon & Sebah, 2003)

$$\zeta(s) = \frac{1}{d_0(1 - 2^{1-s})} \sum_{k=1}^m \frac{(-1)^{k-1} d_k}{k^s} + \gamma_m(s), \tag{4.1}$$

where $d_k = m \sum_{j=k}^m \frac{(m+j-1)! 4^j}{(m-j)! (2j)!}$, we can compute $\zeta(\sigma + it)$ with d decimal digits of accuracy, which requires number of term m roughly equal to $m \approx 1.3d + 0.9|t|$ (Gourdon & Sebah, 2003).

To calculate Eq.(3.4), we need to compute up to $t = N \cdot \Delta t = 4\pi n$, hence we need $m \approx 1.3d + 3.6\pi n$ to obtain the value of $\zeta(\sigma + it)$ with d decimal digits of accuracy, which has expected running time $O(n^2)$.

As the running time to require DFT calculation is $O(N^2)$, thus we need the running time to complete calculations of steps from ① to ④, to be estimated as $O(n^2(\log n)^2)$. Hence it can be seen that primality testing of integer can be conducted in a polynomial time by using this algorithm.

Moreover, we can factor the integer which is composed of two different primes by steps from ① to ④, because the calculated result of $F_n(\omega)$ has only two spectrum according to Theorem 2.

As two spectrum obtained can be given by $\omega_1 = \log q - \log p$ and $\omega_2 = \log q + \log p$ (Musha, 2014), we obtain $p = \sqrt{n \cdot \exp(-\omega_1)}$ and $q = \sqrt{n \cdot \exp(\omega_1)}$ ($q > p$) from them if we let ω_1 is a small spectrum obtained from the calculation of $F_n(\omega)$. From these obtained values for p and q , we have finally to examine whether they satisfy $n = p \cdot q$ or not.

Thus it can be seen that prime factorization for the integer composed of two different primes can be conducted in a polynomial time. There is no efficient, non-quantum integer factorization algorithm is not known now (Yang, 2002), and it has been widely believed that no algorithm is existed that can factor all integers in polynomial time. Thus the presumed difficulty of this problem is at the heart of widely used algorithms in cryptography such as RSA. Contrary to this, we can see that prime factorization for the integer composed of two different primes can be conducted within a polynomial time and it can be shown that this special case belongs to the P class from the theoretical analysis.

However, the validity of this factoring algorithm has been confirmed for only small integers by the restriction of a computer capacity and hence it is necessary to confirm the validity of this algorithm for large integers by using more powerful computer systems.

5. Conclusion

From the spectrum obtained by the Fourier transform of a correlation function generated from the Riemann zeta function, we can see the primality of a integer n if and only the $F_n(\omega)$ has a single spectra for $\omega \geq 0$. Furthermore, it can be shown that the prime factorization can be conducted within a polynomial time for the special case that the integer is composed of two different primes and hence we can conclude that that prime factorization for the integer composed of two different primes is in the P class.

References

- Gourdon, X. and P. Sebah (2003). Numerical evaluation of the Riemann Zeta-function. <http://numbers.computation.free.fr/Constants/Miscellaneous/zetaevaluations.pdf>.
- Kac, M. (1959). *Statistical Independence in Probability Analysis and Number Theory*. The Mathematical Association of America.
- Klee, V. and S. Wagon (1991). *Old and new unsolved problems in plane geometry and number theory*. The Mathematical Association of America.
- Musha, Takaaki (2012). A study on the Riemann hypothesis by the Wigner distribution analysis.. *JP J. Algebra Number Theory Appl.* **24**(2), 137–147.
- Musha, Takaaki (2014). Primality testing and integer factorization by using Fourier transform of a correlation function generated from the Riemann Zeta function. *Theory and Applications of Mathematics & Computer Science* **4**(2), 185–191.
- Shor, P. W. (1997). Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.* **26**(5), 1484–1509.
- Yang, S. Y. (2002). *Number Theory for Computing (2nd Edition)*. Springer-Verlag, New York.
- Yen, N. (1987). Time and frequency representation of acoustic signals by means of the Wigner distribution function: Implementation and interpretation. *The Journal of the Acoustical Society of America* **81**(6), 1841–1850.