UNIVERSITY OF THE
WITWATERSRAND,
JOHANNESBURG

Multidimensional Digital Compression Methods Based On Space-Filling Curves

Conrad J. Haupt

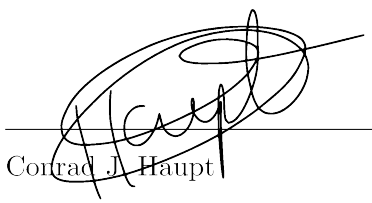
A dissertation submitted to the Faculty of Engineering and the Built Environment, University of the Witwatersrand, Johannesburg, in fulfilment of the requirements for the degree of Master of Science in Engineering.

Johannesburg, August 2021

Declaration

I declare that this dissertation is my own, unaided work, except where otherwise acknowledged. It is being submitted for the degree of Master of Science in Engineering to the University of the Witwatersrand, Johannesburg. It has not been submitted before for any degree or examination to any other university.

Signed this 16th day of August 2021



Conrad J. Haupt

Abstract

Common compression techniques typically do not exploit the underlying structure and dimensionality of the data being processed. However, queries on multidimensional data are already enhanced through the use of spatial data structures and d-dimensional restructuring. Space-Filling Curves (SFCs) are one such technique, mapping the data to the one-dimensional space such that the spatial locality is somewhat preserved. Their application to data compression is not as extensively researched, especially given that such work must be conducted within the context of the type of data being compressed. In this research, the application of SFCs to the compression of Digital Elevation Model and Radio-Astronomy data is evaluated. This has not been extensively researched before, for these types of data. Datasets from the Shuttle Radio Topographical Mission (SRTM) and Square-Kilometre Array telescope (SKA) projects are used in this work. An analysis and discussion of metrics quantifying locality preservation of these curves is provided, to allow for the analysis of how these curves impact the performance of compression schemes for the two aforementioned datasets. It is found that SFCs improve compression ratios in certain cases, the extent to which is dependent on the curve, the data's statistical model, the preprocessing steps, and the metric performance of the curve. Compression and decompression speeds are not as significantly improved, with some combinations of curve and compression scheme giving a reduction in speeds. However, as is predicted by the locality preservation metrics and the data models, the SFCs do contribute to a significant increase in compressibility as measured by the information entropy.

To Lucky 

Acknowledgements

The author would first like to thank his family — Celia, Philip, Rachel, Abigail, Bobbe, and David — for their continued emotional and financial support, as well as their guidance throughout his postgraduate career. He also greatly thanks Sharon for her encouragement and assistance, the author believes without which this dissertation would not be.

The author's sincerest thanks go to Professor Ekow Otoo and Professor Ling Cheng, who supervised him through this work and gave invaluable counsel. He is most grateful for the wealth of knowledge Professor Ekow Otoo shared in the execution of his master's programme, contributing to the core of its thesis. His heartfelt appreciation goes to Professor Ling Cheng for his thorough supervision and advice given during the course of the programme and the creation of the two paper manuscripts and dissertation.

He would also like to thank Jeannine Davids, Mumtaz Adams, and Professor Ken Nixon for their encouragement during his research. They were kind and supportive throughout, playing a major part in the author's academic experience during the programme.

The author appreciates the generosity of the United States Geological Survey in the allowing the free access and use of the Shuttle Radio Topographical Mission data. Secondly, he is most thankful to the South African Radio-Astronomy Observatory, and Selaelo Matlhane for allowing the use of unpublished data from the Square-Kilometre Array telescope.

Contents

List of Figures

List of Tables

Nomenclature

Acronyms

AVX2	Advanced Vector Extensions 2
BMI2	Bit Manipulation Instruction Set 2
BWT	Burrows-Wheeler Transform
DEM	Digital Elevation Model
DPCM	Differential Pulse-Code Modulation
DTM	Digital Terrain Model
EMI	Electromagnetic Interferance
FPGA	Field-Programmable Gate Array
GSFC	Gradient-based Space-Filling Curve
HCR	Hölder Continuity Ratio
HDF5	Hirearchichal Data Format, version 5
HFF	Huffman Encoding
IGRF	Isotropic Gaussian Random Field
kNN	k-Nearest-Neighbour
LZ4	Lempel-Ziv type compression scheme
LZ77	Lempel-Ziv compression scheme from 1977
LZ78	Lempel-Ziv compression scheme from 1978
LZO	Lempel-Ziv-Oberhumer compression scheme
LZW	Lempel-Ziv-Welch compression scheme
ML	Machine Learning

NN	Nearest-Neighbour
NNS	Nearest-Neighbour Stretch
PDR	Partition Diameter Ratio
PR	Partition Ratio
PSR	Partition Surface Ratio
RHCR	Reciprocal Hölder Continuity Ratio
RLE	Run-Length Encoding
RSFC	Recursive Space-Filling Curve
SARAO	South African Radio Astronomy Observatory
SFC	Space-Filling Curve
SFCC	Space-Filling Curve Compression file format
SKA	Square-Kilometre Array
SRTM	Shuttle Radio Topographical Mission
USGS	United States Geological Survey

Space-Filling Curves

$\mathbb{N}_0$	Set of all natural numbers
$\mathbb{M}$	Finite subset of all natural number
$\mathbb{R}$	Set of all real numbers
$\mathbb{Z}$	Set of all integers
$\mathcal{I}$	The unit interval $[0, 1] \in \mathbb{R}$
d	Number of dimensions in a given space
N	Sidelength of a d-dimensional hypercube
N_T	Number of elements in a d-dimensional hypercube of sidelength N

K	Point within a d-dimensional SFC space
$\kappa_{i,l}$	Coordinate for a given point K_i and dimension l
$(\kappa_{i,0}, \kappa_{i,1}, \ldots, \kappa_{i,d-1})$	Coordinates for a given point K_i
x	Index along a given SFC, beginning with zero
$\mathbb{M}_x$	The set of all possible indexes along a given SFC
$\mathbb{M}_K^d$	The set of all possible points within an SFC space
$\mathcal{S}$	An SFC Mapping from one-dimension to d-dimensions
B	Base-of-Recursion for a given Recursive SFC (RSFC)
m	The order of an SFC such that $N_T = B^m$
$\mathcal{S}(x)$	Forward SFC mapping from x to K
$\mathcal{S}^{-1}(K)$	Reverse SFC mapping from K to x
$\mathcal{S}_m^d$	An SFC in d dimensions of order m
S_m^d	A generic Continuous Curve
$\mathcal{P}_m^d$	The Peano Curve
$\mathcal{H}_m^d$	The Hilbert Curve
$\mathcal{Z}_m^d$	The Z-Order Curve (also called the Morton or Lebesgue Curve)
$\mathcal{G}_m^d$	The Gray-Code Curve
$\mathcal{R}_m^d$	The Raster Curve (generated using Row-Major Order)
$\mathcal{A}_m^d$	The $\mathrm{AR^2W^2}$ Curve
$\mathcal{K}_m^d$	The Snake Curve, similar to the Raster Curve
H_m^d	The H-Order/H-Index Curve
$\mathcal{M}_m^d$	The Mitchison Curve
$\mathcal{O}_m^d$	The Onion Curve

$\beta\Omega_m^d$	The $\beta\Omega$ Curve

Data Compression

$\mathbb{D}$	Set of multidimensional data
$\mathbb{D}_f$	Single file from a multidimensional dataset $\mathbb{D}$
$\mathcal{C}$	A generic compression scheme
β	The BitShuffle parameter, 0 or *false* for no BitShuffle and 1 or *true* for BitShuffle to be enabled.
γ	The compression scheme parameter
$\mathcal{R}_c$	Compression ratio of a file for a specific compression scheme
$\mathcal{C}_{\mathcal{S},\beta,\gamma}(\mathbb{D})$	Compression of a dataset $\mathbb{D}$ using SFC $\mathcal{S}$, BitShuffle parameter β, and compression scheme γ
$\mathcal{C}^{-1}_{\mathcal{S},\beta,\gamma}(\mathbb{D})$	Decompression of a dataset $\mathbb{D}$ using SFC $\mathcal{S}$, BitShuffle parameter β, and compression scheme γ
$\mathbb{C}$	A set of compression schemes $\{\gamma_i, \gamma_j, \ldots, \gamma_k\}$

Metrics and Measurements

$\lvert x \rvert$	The absolute value of x
$\lVert \vec{x} \rVert_p$	The p-norm of $\vec{x}$
p	p-norm parameter
q	Locality Ratio normalization exponent
r	Hölder Continuity exponent
δ	Fixed-Distance parameter
l	Dimension Index, such that $0 \leq l < d$

Hölder Continuity Ratio metrics

$g_{p,q,r}(\mathcal{S}, x_i, x_j)$	Generic Hölder Continuity Ratio measurement

$G_{p,q,r}(\mathcal{S})$ Hölder Continuity Ratio metric for any generic aggregation

$G_p^{\max}(\mathcal{S})$ p-normed Maximum Hölder Continuity Ratio metric

$G_p^{\min}(\mathcal{S})$ p-normed Minimum Hölder Continuity Ratio metric

$G_{\mathrm{GL}}^{\min}(\mathcal{S})$ Minimum Hölder Continuity Ratio metric by Gotsman and Lindenbaum

$G_{\mathrm{GL}}^{\max}(\mathcal{S})$ Maximum Hölder Continuity Ratio metric by Gotsman and Lindenbaum

Partition Ratio metrics

$P(x_i, x_j)$ Partition of all points and indices bounded by x_i and x_j

$\mathrm{surf}(\mathcal{S}, x_i, x_j)$ Surface Area of partition $P(x_i, x_j)$ along $\mathcal{S}$

$\mathrm{diam}(\mathcal{S}, x_i, x_j)$ Diameter of partition $P(x_i, x_j)$ along $\mathcal{S}$

$g^{\mathrm{surf}}(\mathcal{S}, x_i, x_j)$ Partition Surface Ratio measurement

$g^{\mathrm{diam}}(\mathcal{S}, x_i, x_j)$ Partition Diameter Ratio measurement

$G_{\max}^{\mathrm{surf}}(\mathcal{S})$ Maximum Partition Surface Ratio metric for SFC $\mathcal{S}$

$G_{\mathrm{avg}}^{\mathrm{surf}}(\mathcal{S})$ Average Partition Surface Ratio metric for SFC $\mathcal{S}$

$G_{\max}^{\mathrm{diam}}(\mathcal{S})$ Maximum Diameter Partition Ratio metric for SFC $\mathcal{S}$

$G_{\mathrm{avg}}^{\mathrm{diam}}(\mathcal{S})$ Average Diameter Partition Ratio metric for SFC $\mathcal{S}$

Reciprocal Hölder Continuity Ratio metrics

$f_{p,q,r}(\mathcal{S}, x_i, x_j)$ Generic Reciprocal Hölder Continuity Ratio measurement

$F_{p,q,r}(\mathcal{S})$ Reciprocal Hölder Continuity Ratio metric for any generic aggregation

$F_p^{\mathrm{PKK}}(\mathcal{S})$ Weighted-Sum/Reciprocal Hölder Continuity Ratio

$f_{q,\delta}^{\mathrm{edge}}(\mathcal{S}, x_i, x_j)$ EdgeSum measurement

$F_{q,\delta}^{\mathrm{edge}}(\mathcal{S})$ EdgeSum Metric

$f^{\log}(x_i, x_j)$ Logarithmic EdgeSum measurement

$F_{\max}^{\log}(\mathcal{S})$	Maximum Logarithmic EdgeSum metric
$F_{\text{avg}}^{\log}(\mathcal{S})$	Average Logarithmic EdgeSum metric
$f_{\text{avg}}^{\text{nns}}(\mathcal{S}, x_i)$	Average Nearest-Neighbour Stretch measurement
$f_{\max}^{\text{nns}}(\mathcal{S}, x_i)$	Maximum Nearest-Neighbour Stretch measurement
$F_{\text{avg}}^{\text{nns}}(\mathcal{S})$	Average-Average Nearest-Neighbour Stretch metric
$F_{\max}^{\text{nns}}(\mathcal{S})$	Average-Maximum Nearest-Neighbour Stretch metric

Description and Irregularity Vector metrics

V	Any given Description Vector
V_l	Description Vector in dimension l
V_T	Total Description Vector, for all dimensions ($\forall 0 \leq l < d$)
$T_l(\mathcal{S}, N, l)$	Number of generic segment type T in SFC $\mathcal{S}$ with sidelength N for dimension l
$T_T(\mathcal{S}, N, l)$	Total Number of a generic segment type T in d-dimensional SFC $\mathcal{S}$ with sidelength N
$J_l(\mathcal{S}, N, l)$	Number of jump segments in dimension l
$C_l(\mathcal{S}, N, l)$	Number of contiguity segments in dimension l
$R_l(\mathcal{S}, N, l)$	Number of reverse segments in dimension l
$F_l(\mathcal{S}, N, l)$	Number of forward segments in dimension l
$S_l(\mathcal{S}, N, l)$	Number of still segments in dimension l
$C_T(\mathcal{S}, N, d)$	Total number of contiguity segments in SFC $\mathcal{S}$
$F_T(\mathcal{S}, N, d)$	Total number of forward segments in SFC $\mathcal{S}$
$J_T(\mathcal{S}, N, d)$	Total number of jump segments in SFC $\mathcal{S}$
$R_T(\mathcal{S}, N, d)$	Total number of reverse segments in SFC $\mathcal{S}$

$S_T(\mathcal{S}, N, d)$ Total number of still segments in SFC $\mathcal{S}$

$I_l(\mathcal{S}, N, l)$ Number of Irregularities in dimensions l for SFC $\mathcal{S}$ with sidelength N

$I_T(\mathcal{S}, N, d)$ Total number of Irregularities for the d-dimensional SFC $\mathcal{S}$ with sidelength N

$V_I(\mathcal{S}, N, d)$ Irregularity Vector for the d-dimensional SFC $\mathcal{S}$ with sidelength N

Bounding-Region metrics

$WL_p(\mathcal{S})$ The Worst-case p-norm distance measure for SFC $\mathcal{S}$, the same as $G_p^{\max}$

$WL_1(\mathcal{S})$ The Worst-case 1-norm distance measure

$WL_2(\mathcal{S})$ The Worst-case 2-norm distance measure

$WL_\infty(\mathcal{S})$ The Worst-case ∞-norm distance measure

$\mathrm{bbox}(K_i, \ldots)$ The Bounding-Box encapsulating points K_i

$\mathrm{boct}(K_i, \ldots)$ The Bounding-Octagon encapsulating points K_i

$\mathrm{bball}_p(\,\cdot\,)$ The p-normed Bounding-Ball encapsulating points K_i

$\mathrm{per}(\mathrm{bbox}(\,\cdot\,))$ Perimeter of a Bounding-Region, such as $\mathrm{bbox}(\,\cdot\,)$

$\mathrm{surf}(\mathrm{bbox}(\,\cdot\,))$ d-dimensional Surface Area of a Bounding-Region, such as $\mathrm{bbox}(\,\cdot\,)$

$|\mathrm{bbox}(\,\cdot\,)|$ d-dimensional Volume of a Bounding-Region, such as $\mathrm{bbox}(\,\cdot\,)$

WBA Worst-Case Bounding-Box Area

WBP Worst-Case Bounding-Box Perimeter

TBA Worst-case Total Bounding-Box Area

ABA Average Total Bounding-Box Area

ABP Square Average Relative Total Bounding-Box Perimeter

WOA Worst-case Bounding-Octagon Area

AOA Average Total Bounding-Octagon Area

WOP	Worst-case Bounding-Octagon Perimeter
WD_p	Worst-case p-normed Diameter Ratio
WBB_p	Worst-case p-normed Bounding-Ball Volume
WS	Worst-case Surface Ratio
WBV	Worst-case Bounding-Box Volume Ratio
WBS	Worst-case Bounding-Box Surface Ratio

Clustering metric

r_s	The region size parameter
R	An arbitrary d-dimensional region with size-parameter r_s
R_i	An arbitrary position of R in the SFC space ($R_i \in R$)
$\square$	A square or hypercubic region ($r_s \times \cdots \times r_s$)
$\square$	A rectangular or hyperrectangular region ($\alpha_0 r_s \times \cdots \alpha_{d-1} r_s$)
$\bigcirc$	A circular or hyperspherical region, with radius r_s
$\bigcirc$	An octogonal region of width r_s
λ	A specific rotation for a region R
Λ^*	The set of all possible rotations λ
Λ	A set of rotations $\Lambda \subseteq \Lambda^*$
$\eta(\mathcal{S}, R_i)$	The number of clusters for a given region R_i and SFC $\mathcal{S}$
$H_{\text{avg}}(\mathcal{S}, R)$	The Average Clustering metric: the average number of clusters for all positions of region R and the SFC $\mathcal{S}$
$H_{\text{max}}(\mathcal{S}, R)$	The Maximum Clustering metric: the maximum number of clusters for all positions of region R and the SFC $\mathcal{S}$

List of Publications

1. **C. J. Haupt**, E. J. Otoo, and L. Cheng, "Effect of d-Dimensional Re-orderings on Lossless Compression of Radio-Astronomy and Digital Elevation Data," *IEEE Access*, vol. 9, pp. 80 415–80 433, 2021, ISSN: 2169-3536. DOI: 10.1109/ACCESS.2021.3084838. [Online]. Available: https://ieeexplore.ieee.org/document/9443202.
2. **C. J. Haupt**, E. J. Otoo, and L. Cheng, "A review of metrics quantifying locality preservation of Space-Filling Curves and their applications," To be Submitted to the IEEE Transactions on Computational Imaging, 2021.

Chapter 1

Introduction

The way in which multidimensional data is stored is important to the functioning of certain algorithms as the number of dimensions has a significant impact on data access and computational performance. Some algorithms are explicitly defined for multidimensional data; such as k-Nearest Neighbour searches, database queries, and much of scientific computing. In most of these scenarios, the method by which the data are mapped to and stored in memory is the driving factor behind the performance of an algorithm, resulting in multidimensional specific data structures such as R-Trees. The mapping from d dimensions to one dimension is an important part of this process as high-dimensional datasets have to be stored in the one-dimensional memory address space. There are multiple techniques for carrying out this mapping process, with one such method being the focus of this dissertation: Space-Filling Curves (SFCs).

These curves are mappings between the one-dimensional space and any number of dimensions d [1]. In their *true* mathematical form, they are infinitely refined curves that fill the d-dimensional space in its entirety. Multidimensional algorithms have benefited through the use of finite variants of these SFCs [2]–[4]. In many of these cases, the SFCs are used to *better* traverse the d-dimensional space on which these algorithms are applied. Given that the curves fill the space, they typically traverse local contexts before those farthest away. This property is referred to as locality preservation: keeping points that are nearby in the d-dimensional space nearby in the one-dimensional space. For multidimensional algorithms, that work within the d-dimensional space, having neighbouring values be nearby in memory can improve performance by reducing the computational cost [5] and increasing hardware efficiencies through cache hits [6], [7]. Much research has been conducted on how to apply SFCs to these algorithms [8]–[11].

Utilizing the same concept of spatial data structures — storing locally correlated data together — one can assume that restructuring multidimensional data using

SFCs should increase its compressibility. For compression techniques that use local contexts to reduce the size requirements, this increase in compressibility can be exploited. This concept has been explored before, with varying results and limited types of datasets [12]–[15]. However, compression is always within the context of the data. Though the literature is extensive, there are still areas where the concept of SFC-based compression has not been extensively researched; specifically Digital Elevation Model (DEM) and Radio-Astronomy data. This is the core contribution of the work presented here.

An additional focus of this work is in the analysis and discussion of metrics to rank these curves based on their locality preservation, for specific applications. These metrics are important in identifying how different properties arise from various SFCs and how they are best matched with applications. If there is an appropriate metric for a given application, it can be used to find an optimal SFC mapping to improve the performance of the computational problem. Many metrics quantifying locality preservation are found in the literature, but they have not been extensively compared. Such an analysis is given in this dissertation. The compression of DEM and Radio-Astronomy data presented here utilizes the accompanying contributions, related to these metrics, in the analysis of which SFCs perform best for the two types of data.

1.1 Problem Statement and Research Questions

There is a gap in the application of SFCs to data compression for DEM and Radio-Astronomy data. Furthermore, existing research on SFC-based compression methods for other sources of data typically do not compare the same compression scheme with different SFCs. This methodology does not explore the effect of SFCs on data compression but instead shows that a bespoke compression scheme, incorporating SFCs, can outperform other non-SFC based compression techniques. Conclusions made in the literature indicate that the type of data being compressed is a driving factor behind the effect of SFC reorderings on compression performance. Two such types of data to which SFC-based compression has not been extensively applied

are Digital Elevation Model (DEM) and Radio-Astronomy Data. This leads to the following three research questions, which are addressed by this dissertation.

What metrics can be used to rank SFCs for a specific computational task? Given that there are numerous SFCs, with their own properties, it is natural to ask if there is a way of choosing one over another for a specific computational application.

What is the impact of SFC reorderings on the compressibility of DEM and Radio-Astronomy data? The difference between compression performance and compressibility is important as the former may not be able to eliminate the redundancies indicated by the latter.

What is the impact of SFC reorderings on compression performance for DEM and Radio-Astronomy data? Given that these types of data have not been extensively researched before, within the context of SFC-based compression, it is necessary to understand their impact on compression performance to further the understanding of SFC-based compression methods. Compression performance includes compression ratios and compression and decompression speeds.

1.2 Research Contribution and Significance

The work presented here contributes to the overall picture of SFC reorderings, methods to compare them, and their application to the compression of DEM and Radio-Astronomy data. As previously mentioned, SFCs are appropriate for multidimensional and spatial algorithms as they have the property of locality preservation. However, this is not a well-defined term and thus is up for interpretation. In this dissertation, an investigation and comparison of metrics that quantify locality preservation for specific applications is given. Though these metrics come from research published in the literature, an analysis comparing different types has not been made before. In covering these metrics, the relevance of their definitions to specific

applications is made. More specifically, this contribution is in the structuring and presentation of existing locality preservation metrics, the identification of relationships between and classification of metrics and their measurements, the discussion of the performance of different SFCs for various metrics, and the discussion of appropriate applications for each metric.

The core contributions of this work are in the application of SFC reorderings to the compression of DEM and radio-astronomy data. Though other datasets have been compressed using SFCs under similar scenarios, these types of data have not been covered to this extent before. DEM data has applications in environmental research [16], [17] and urban planning [17], [18]. Therefore, the compression of such data is important in reducing the storage and transfer costs for a broad range of fields and contexts. Better compression schemes for DEM data can reduce the storage and transfer costs — both financial and computational — for these applications. The prevalence of such data makes the impact of better DEM compression significant.

In contrast to DEM data, radio-astronomy data is not as common place. The specific dataset utilized in this work is from the Square-Kilometre Array (SKA) project, soon to be the largest radio-telescope in history [19]. The quantity of data produced by this array is of the order of petabits-per-second, creating a problem of transport and storage [19]. Any method that can reduce the data storage requirements of this process would greatly benefit the operation of the array. This data is inherently multidimensional and thus a good candidate for SFC-based preprocessing.

The work presented here shows that the extent to which DEM and radio-astronomy data can be compressed, and the impact of different orderings on the compression performance, is highly dependent on the underlying statistics of the data. Furthermore, any method that exposes highly redundant components of the data improves the compressibility of the datasets.

1.3 Scope and Research Objectives

The work presented here focuses on SFC reorderings for the compression of DEM and Radio-Astronomy data. The aim of this research is to evaluate the effect of these curves on the compression performance within this context. A system to compress and measure certain compression performance metrics is presented, which is then used to achieve this aim.

The first objective of this research is the identification, comparison, and analysis of metrics to rank SFCs for a specified application. Alongside this objective is the development of a methodology to compare these metrics: the measurement-aggregate-metric method. Achieving this objective enables further research into locality preservation for multidimensional computational problems. The knowledge produced here is used to analyse results produced by later objectives.

A number of objectives are necessary to achieve the primary aim of SFC-based compression. The first of these is the identification and analysis of two DEM and Radio-Astronomy datasets. The data are sourced from the Shuttle Radio Topographical Mission (SRTM) and Square-Kilometre Array (SKA) project. A statistical analysis of both datasets is conducted to determine how the underlying values behave in the d-dimensional spaces. Their data types and spatial correlations, along with the labelling of each dimension, are identified in this analysis. The identification of these datasets and their analysis is given in Section 2.4.

The third overall objective is the development of a system to transform the two datasets using SFCs, compress and decompress the resulting data, and measure the compression ratio and speed achieved. This takes the form of a `C++` program developed for this research, which allows for any combination of SFC, BitShuffle (a preprocessing step), and compression scheme. These combinations are parameterized to allow the analysis to fully evaluate the effect a given component has on compression performance. This is to avoid the situation in some literature on SFC-based compression, where the impact of SFCs on compression cannot be separated from the compression scheme used. The initial compression performance for the SKA dataset was insufficient. Therefore, BitShuffle — a bitwise preprocessing

step — was incorporated into the research scope to increase compression ratios [20]. Given that the implemented software was designed with good software practices and object-oriented programming methodologies, BitShuffle was integrated for both datasets.

To ensure that the objectives were attainable, the SFCs implemented in the compression program were limited to those that are most popular in the literature: the Z-Order, Gray-Code, and Hilbert Curves, and the standard Raster Curve/Scan. Within tools that manage the chosen datasets, being DEM file formats and HDF5 [21], lossless compression schemes are the most common. Therefore, to limit the scope of this research, only lossless compression schemes are covered in this research. For more on the compression schemes used and BitShuffle, see Section 2.5.

The research questions explicitly state compressibility and compression performance as separate components of the analysis. The fourth and fifth objectives of this research are in the analysis of these components, respectively. The compressibility of the two datasets is determined using the SFC-mapped and preprocessed data, without the final compression step. The principal method to evaluate compressibility is the information entropy of the processed dataset files. Given that the total entropy of a file is independent of the ordering of its values, the block-entropy is used to determine how the restructuring of the data impacts the local information entropy.

The compression performance is evaluated using the compression ratio and compression and decompression speeds obtained through the use of the third objective's `C++` program. This is important as the performance of a compression scheme is not only in by how much it reduces the storage requirements of a dataset, but also in how fast it can achieve this. By analysing the impact of different combinations of SFCs and compression schemes on the performance achieved, relationships between the statistical behaviour of the datasets, the locality preservation of the curves, and the underlying techniques used by the compression schemes are identified. The results of this analysis inform those applying SFCs to a computing system on how well the mapped data may be compressed, as well as the potential impact on computational performance.

1.4 Structure of the Dissertation

This document is structured using a paper format, with two publications resulting from this research in Chapters 3 and 4 respectively. The paper given in Chapter 4 has been published in the IEEE Access journal as of the 28th of May 2021 [22]. The literature review and background content for this research is given in Chapter 2. The fundamentals of Space-Filling Curves, their mathematical definition, and their application to compression and computing are given in Sections 2.1, 2.1.1 and 2.1.3 respectively. Section 2.2 contains an explanation of locality preservation, how it differs from order and distance preservation, and why it is used to evaluate SFCs.

Background on metrics, and the measurement-aggregate-metric methodology used to compare them, is provided in Section 2.3. The two datasets used to evaluate the effect of SFC reorderings on compression are discussed and analysed in Section 2.4. This includes a statistical analysis of the data, fulfilling the second objective from Section 1.3. The identification and setup of the chosen compression schemes and the preprocessing scheme BitShuffle are given in Section 2.5. The compression algorithms and preprocessing schemes that make up each scheme are also provided.

Chapter 3 contains an analysis of locality preservation metrics for SFCs, constituting the first publication from this work. The metrics are grouped into types and subtypes (Sections 3.1 and 3.2), associated SFC rankings are analysed utilizing analytical and empirical bounds on metric values (Section 3.3), and the relationship between applications and the metrics' definitions are analysed (Section 3.4).

The core contribution of this dissertation, the compression of DEM and radio-astronomy data with SFCs, is given in the second paper in Chapter 4. The third, fourth, and fifth objectives are achieved through the analysis presented in this paper. The methodology by which results are obtained is given in Section 4.1, with their analysis in Section 4.2.

Lastly, Section 5.1 summarizes the contribution of this research, with Sections 5.2 and 5.3 covering recommendations and topics future research, and the conclusion to the three research questions respectively.

Chapter 2

Background

2.1 Space-Filling Curves

The first Space-Filling Curve (SFC) was discovered by Peano [23] in 1890 after Cantor proved that the cardinality of a d-dimensional unit-hypercube is the same as the unit-interval: $|\mathcal{I}| = |\mathcal{I}^d|$ [24]. Cantor's result shows that a continuous curve can *fill* a hypercube in its entirety using a surjective mapping [1], [25]. Hilbert [26] later found a second SFC, now referred to as the Hilbert Curve, which is more common in literature on the topic of SFCs. Lebesgue discovered the Lebesgue curve by interleaving the bits of d integer coordinate values resulting in an observable zigzag pattern [27]. It was popularized by Morton in his work applying it to geodetic databases [28] and has been referred to as the Morton Curve. We refer to is as the Z-Order Curve [29]. After Peano [23] and Hilbert's [26] discoveries, other curves have been identified; such as the Gray-Code [29], $\beta\Omega$ [30], AR^2W^2 [31], H-Order [32], and Onion [33] curves.

These curves wrap around themselves and their local contexts before traversing into further regions of the d-dimensional space. This property implies that d-dimensional neighbouring points are likely to be *close* in the one-dimensional SFC-mapped space. Depending on the SFC used, the distance to a d-dimensional neighbour in the one-dimensional space may be less than with the Raster Scan on average or in the worst case [34]. This is not always the case as each curve has its own characteristics and structure. Significant research exists on evaluating which SFCs are best at this *locality preservation* process (see Chapter 3) [2], [35], [36].

The Gray-Code Curve is created by treating the Z-Order one-dimensional index as a Gray Code, converting it to a Gray Code index, and then conducting the Z-Order Curve mapping on the resulting value [37]. The Raster Scan is technically an SFC; however, as it is used as the ordering for most data on disk and in memory, it is a

trivial example and is instead used as the reference curve in this and other research. All SFCs discussed in this work are illustrated in Fig. 2.1 with the covering of the general definition of SFCs.

A *true* SFC $\mathcal{S}$ is defined as a surjective mapping from the unit-interval $\mathcal{I} = [0, 1]$ to the unit-hypercube $\mathcal{I}^d = [0, 1]^d$ in d dimensions. This is a direct result of Cantor's work [24] in 1878 showing that the cardinality of the set of real-numbers $\mathbb{R}$ is the same in one dimension as well as d dimensions ($|\mathbb{R}| = |\mathbb{R}^d|$) [25]. An SFC is a curve that passes through all d-dimensional points within $\mathcal{I}^d$ and is defined as the ordering of points by the mapping $\mathcal{S} : \mathcal{I} \to \mathcal{I}^d$. Such a mapping can be bijective but cannot also be continuous, as shown by Netto [38]. Therefore, most SFCs are surjective to maintain the continuous constraint. For computational purposes, we must take a finite solution of $\mathcal{S}$ to apply it to data, giving a bijective mapping whether the *true* SFC is continuous or not. The following definition describes an SFC appropriate for computation with an illustration of how their surjective variants can be obtained.

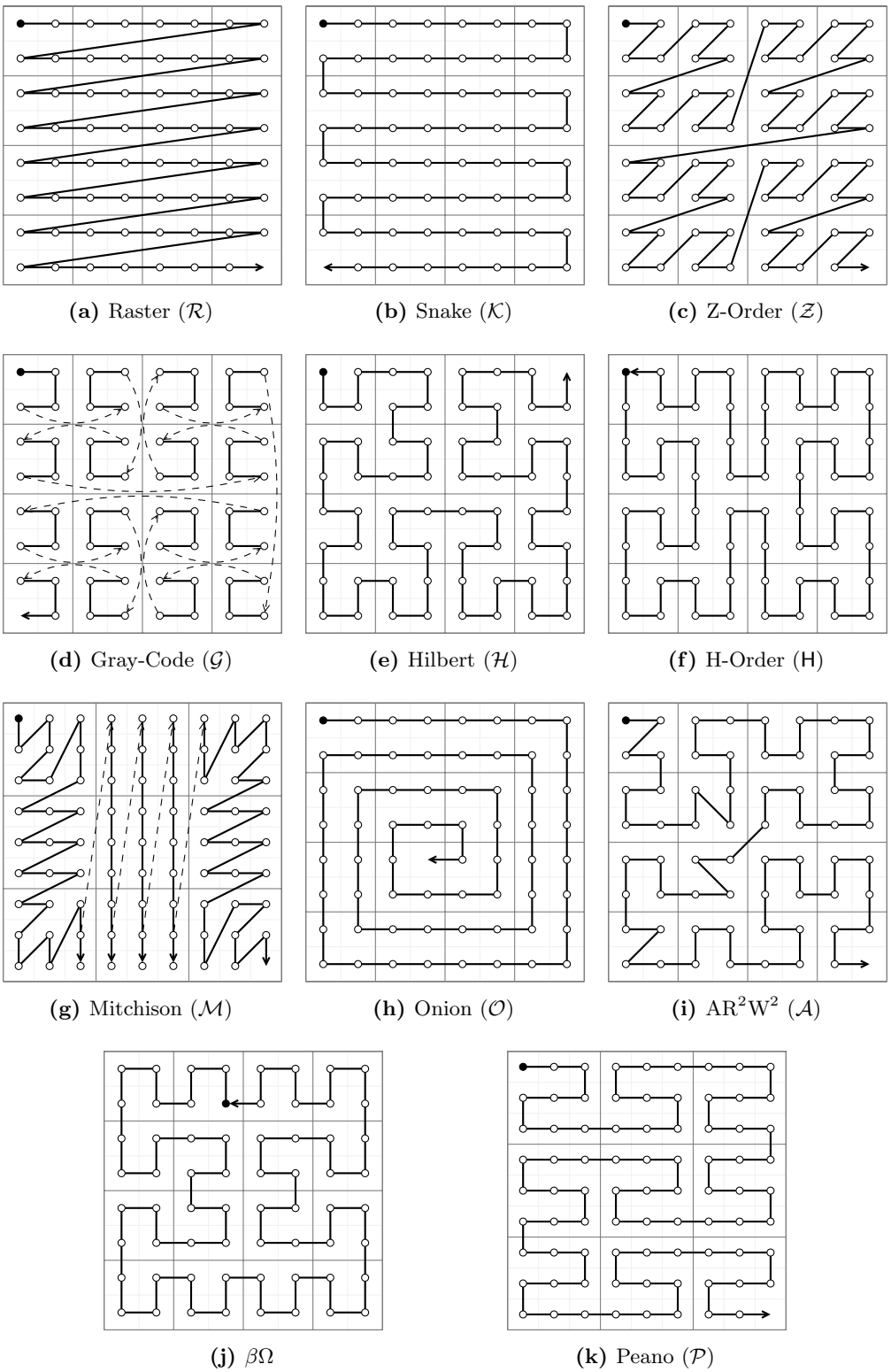
2.1.1 Definition

Let $\mathbb{M}$ be an arbitrary finite subset of the set of natural numbers $\mathbb{N}_0$, with d-dimensional variants $\mathbb{M}^d$ and $\mathbb{N}_0^d$ respectively. We define a finite SFC mapping $\mathcal{S}$ as

$$\mathcal{S} : \mathbb{M}_x \mapsto \mathbb{M}_K^d, \tag{2.1}$$

where $\mathbb{M}_x$ is the set of all one-dimensional indices $\{x_i, x_j, \ldots\}$ and $\mathbb{M}_K^d$ is the set of all d-dimensional points $\{K_i, K_j, \ldots\}$. The cardinalities of the two sets are equal by definition: $|\mathbb{M}_x| = |\mathbb{M}_K^d|$. This is also the case for the surjective *true* SFCs. We define the size of these sets N_T as follows:

$$|\mathbb{M}_x| = N_T = N^d = |\mathbb{M}_K^d|. \tag{2.2}$$



(a) Raster ($\mathcal{R}$) **(b)** Snake ($\mathcal{K}$) **(c)** Z-Order ($\mathcal{Z}$)

(d) Gray-Code ($\mathcal{G}$) **(e)** Hilbert ($\mathcal{H}$) **(f)** H-Order (H)

(g) Mitchison ($\mathcal{M}$) **(h)** Onion ($\mathcal{O}$) **(i)** $\mathrm{AR}^2\mathrm{W}^2$ ($\mathcal{A}$)

(j) $\beta\Omega$ **(k)** Peano ($\mathcal{P}$)

Figure 2.1: Two-dimensional 3rd-order illustrations of some popular Space-Filling Curves. The markers represent points in the SFC-space with the solid marker indicating the first point along the SFC. Symbols denoting each respective SFC are given in the sub-captions.

Many SFCs are constrained to a hypercube, and thus $N_T = N^d$. However, this need not be the case, as with the Balanced-GP Curve by Haverkort and van Walderveen [2]. Some curves are Recursive Space-Filling Curves (RSFCs), with their *true* definitions being at the limit as their recursion depth or order m tends to infinity. An RSFC will subdivide into B subregions for each incremental increase in the order m. The *true* surjective RSFC is obtained with $\lim_{m\to\infty} B^m = N_T$.

This definition for an RSFC is not as strict as some in the literature, where the B new subregions must be equivalent to the same RSFC at a lower order [31]. However, the idea of an SFC being recursive is important as the self-similarity between quadrants can assist the analytical process, as is seen in the literature [1]. Moreover, RSFCs fill the entire subregion before traversing to other regions of the SFC space, a property identified by Lempel and Ziv in their work on the compression of two-dimensional data using the Hilbert Curve [39].

The SFC mapping $\mathcal{S}$ transforms the index $0 \leq x_i \leq N_T - 1$ to a point $K_i = (\kappa_0, \kappa_1, \ldots, \kappa_{d-1})$ where $0 \leq \kappa_i \leq N_T - 1$. The SFC index is sometimes referred to as an SFC distance though this is only geometrically applicable in the one-dimensional SFC space; however, the SFC index and distances have the same values.

The number of regions B is called the base-of-recursion and is dependent on the number of dimensions d. The depth-of-recursion m is also referred to as the SFC order, allowing us to refer to a specific finite SFC as an mth order d-dimensional SFC. The SFCs shown in Fig. 2.1 are 3rd-order ($m = 3$) two-dimensional ($d = 2$) SFCs. The Z-Order, Gray-Code, Hilbert, H-Order, $\mathrm{AR^2W^2}$, $\beta\Omega$, and Peano Curves are recursive. Though the Mitchison and Onion Curves may appear recursive in nature, they are not. Therefore, the Raster, Snake, Mitchison, and Onion Curves illustrated are not the 3rd-order variants but instead the variants which encompass an equivalent space to the others. Additionally, the Peano Curve has a different base-of-recursion to the other RSFCs illustrated here. Whereas the rest have $B = 4$, the Peano Curve has $B = 9$; for two-dimensions.

For an RSFC constrained to a hypercube, we can define the sidelength N as

$$N = B^{\frac{m}{d}} = 2^m, \tag{2.3}$$

implying that $\left|\mathbb{M}_K^d\right| = |\mathbb{M}_x| = 2^{md}$. As is obvious, the Peano Curve would replace 2 with 3. However, in some cases the base-of-recursion cannot be easily split by the number of dimensions, as is the case with the Balanced-GP Curve [2]. Each increase in the order m divides a different dimension of the Balanced-GP Curve into three ($B = 3$). This is unlike other RSFCs shown here, where division into B regions is irrespective of the dimension, i.e., the Hilbert Curve divides its d-dimensional space in two for each dimension. Furthermore, the Balanced-GP Curve is constrained to a hyperrectangle, indicating that a sidelength is not always an appropriate measure of the size of the SFC space. Therefore, the total number of points on the curve N_T is important as it is independent of the shape of the SFC.

Regardless of whether the SFC is recursive or constrained to a hypercube, or neither, the resulting finite SFC mapping transforms the set of N_T one-dimensional positive-integer indices to the set of d-dimensional coordinates. The final SFC mappings are shown below in (2.4) and (2.5).

$$\mathcal{S}\colon \mathbb{M}_x \mapsto \mathbb{M}_K^d \tag{2.4}$$

$$\mathcal{S}^{-1}\colon \mathbb{M}_K^d \mapsto \mathbb{M}_x \tag{2.5}$$

It is useful to define functions that apply this mapping to input values. For a given index x and its corresponding d-dimensional point K, the functions that map between them are given as

$$\mathcal{S}(x) = K = (\kappa_0, \ldots, \kappa_{d-1}) \tag{2.6}$$

and

$$\mathcal{S}^{-1}(K) = x, . \tag{2.7}$$

In some instances, equations in this document do not explicitly denote the usage of these SFC mapping functions, to reduce typesetting space requirements. In all cases, the subscript variable for indices x_i and points K_i identify index-point pairs obtained through the use of (2.6) and (2.7). The constraints on x and κ, for a given SFC, are

$$0 \leq \kappa_i < \begin{cases} B^{\frac{m}{d}} & \mathcal{S} \text{ is an RSFC in a hypercube} \\ N & \mathcal{S} \text{ is in a hypercube} \\ N_i & \text{otherwise} \end{cases} \tag{2.8}$$

and

$$0 \leq x < \begin{cases} B^m & \mathcal{S} \text{ is an RSFC} \\ N_T & \text{always} \end{cases} \tag{2.9}$$

There are also continuous SFCs, which have different properties to non-continuous SFCs [40]. By the strictest of definitions, all SFCs are continuous as this is a requirement to be a curve [1]. Though non-continuous curve representations, such as the Z-Order Curve, can be made to be continuous through a change of sets [1, p. 97], this is not the same usage of *continuous* as with continuous SFCs. For a given SFC to be continuous, each adjacent point-pair must be separated by a Manhattan Distance of 1:

$$\|K_{i+1} - K_i\|_1 = 1, \quad 0 \leq i < N_T. \tag{2.10}$$

Note that $\|\cdot\|_1$ denotes the Manhattan distance between points K_i and K_j. This is equivalent to the p-norm with $p = 1$, though more on this is given in Chapter 3. Using (2.10), the Raster, Z-Order, Gray-Code, Mitchison, and AR^{2}W^2 Curves are not continuous SFCs. Some curves are cyclic, which is when their first and last indices are adjacent in the d-dimensional space [32]. The condition for an SFC to be cyclic,

$$\|K_{N_T-1} - K_0\|_2 = 1, \tag{2.11}$$

uses the Euclidean Norm and not the Infinity Norm, implying that a cyclic SFC may also be continuous [32]. Cyclic SFCs allow one-dimensional distances or index-differences to be computed with modulo N_T, as is the case with the H-Order Curve in the work by Niedermeier, Reinhardt, and Sanders [32].

2.1.2 Applications to Compression

In 1986, Lempel and Ziv [39] showed that the compressibility of an image is lower-bounded by the same image when ordered using the Hilbert Curve [26]. They describe this behaviour using their encoder published in 1978, commonly referred to as LZ78 [41], [42]. However, their conclusion is not universal as it is dependent on the type of encoder defined in their work, one that is derived from LZ78. There are a few published cases where the use of SFCs are used to effectively compress data, and in some cases outperform the standard Raster Scan. Pinciroli *et al.* use the Hilbert Curve to compress angiocardiographic images to a compression ratio of approximately $\mathcal{R}_c = 6$, though lossy in nature [13]. Another application, also to medical images, is the work by Liang *et al.*, where Run-Length Encoding (RLE) [43], LZW [44], LZ77 [41], and Huffman Encoding (HFF) [45] are applied to CT Scan images with differential coding and reordering using the Hilbert Curve [14]. They achieve a maximum compression ratio of 3.42 utilizing LZW and HFF, with differential coding and Hilbert Curve reordering as preprocessing steps.

An application of SFCs to the compression of colour images is shown by Abdollahi *et al.* [15] where they combine Move-to-Front Encoding with various SFCs as their compression technique. They quantify the effect on compressibility, assuming an entropy-encoder such as HFF, using the entropy-ratio instead of $\mathcal{R}_c$: the ratio between the resulting image and input image's entropies. For all images they investigated, Abdollahi *et al.* achieved the lowest entropy-ratios when the Hilbert Curve was applied. Provine and Rangayyan compress greyscale images mapped with the Hilbert and Raster Curves using HFF, Arithmetic Coding, Predictive Coding, and LZW [46]. The highest compression ratios they achieve, of approximately $\mathcal{R}_c = 5$, are obtained with HFF combined with differential coding, a form of predictive coding.

In some cases, bespoke SFCs have been designed to assist with the compression of data. Ouni, Lassoued, and Abid develop a *gradient-based* SFC (GSFC) for the compression of two-dimensional images [47], [48]. When used in conjunction with the Graphics Interchange Format (GIF) [49], which uses LZW, their *vote* variant of the GSFC achieves better compression than with the Raster or Hilbert Curves [47]. However, with the PNG format [50], which uses DEFLATE [51] (a combination

of LZ77 and HFF), all SFCs are approximately the same: the Hilbert Curve and GSFC achieve similar compression ratios [47]. When compared to other compression schemes, the lossless JPEG compressor JPEG-LS nearly always achieves better compression than GIF, PNG, and their Differential Pulse Code Modulation (DPCM) [52], GSFC, and HFF based technique [48]. Nevertheless, their GSFC-based compression technique achieves compression ratios just above those for JPEG-LS, indicating that further work may result in superior performance.

Scarmana and McDougall apply SFCs to the compression of Digital Terrain Models (DTMs), with results showing the fastest and best compression when the Hilbert Curve is used with DPCM [12]. They compare the compression performance of the Hilbert Curve and DPCM to LZW, JPEG2000, JPEG-LS, and DEFLATE with the Raster Scan. However, they do not compare the results to DPCM with any other orderings or SFCs. Furthermore, they only investigate three DTM files.

Even though Pinciroli *et al.* [13], Liang *et al.* [14], Abdollahi *et al.* [15], and Scarmana and McDougall [12] achieve larger compression ratios and lower entropies when utilizing the Hilbert Curve compared to the standard Raster Scan, they do not compare SFCs to the Raster Scan within the same compression schemes. Provine and Rangayyan do compare two SFCs, but one is the *base-line* Raster Curve [46]. Ouni, Lassoued, and Abid do compare the Raster Scan with two SFCs, but not all combinations of SFCs and compression schemes are explored [47], [48]. Additionally, literature covering the theoretical performance of SFCs, within the context of data compression, differ in their conclusions on the benefit of using SFCs for d-dimensional data reordering. Quin and Yanagisawa show analytically that for images made of artificial random ellipses, the use of SFC-reorderings — before a subsequent RLE compression step — does not improve the compressibility of the data [53]. They come to this conclusion by measuring the length of runs when using RLE, with longer lengths being preferable. For generic images and lossless context-based compression techniques, Memon, Neuhoff, and Shende calculate the theoretical predication gains when utilizing different ordering techniques and conclude that there is no significant prediction gain when utilizing the Hilbert Curve over the Raster Scan [54]. They conjecture that the Raster Scan may result in better compression performance over SFCs because common compression schemes are either explicitly or implicitly

designed for row-major order structured data. Whether focusing on empirical results or theoretical results, it is easily argued that existing literature does not contain the full picture on compression performance with SFC-based reorderings.

For compression of DEM data without the use of SFCs, Rane and Sapiro achieve an average compression ratio of $\mathcal{R}_c = 11.7$ with JPEG-LS [55]. The DEM data they use contains 1201×1201 16-bit integer pixels. They find that the bit-level redundancies in DEM integer data can be exploited by compression techniques, specifically JPEG-LS, if the elevation values are restructured. For example, they show that treating a 24 bit integer as two separate two and one byte integers results in better compression when using JPEG-LS; owing to the higher redundancy of the most significant two bytes.

Compression of astronomy data is dependent on the data type used, as is also the case with DEM data. Masui *et al.* develop a bit-level reordering scheme to compress radio-astronomy data [20]. Their final compression scheme BitShuffle restructures values into groups of bits by their underlying significance, resulting in higher compression ratios with compression schemes that operate on multi-byte symbols. Though they refer to BitShuffle as the preprocessing step and the subsequently applied LZ4 [56] compression scheme, it is more appropriate at times to refer to only the preprocessing step as BitShuffle [57]. Though they apply the technique to integer data, it is also applicable to floating-point data. Zheng *et al.* implement a lossless compression technique for integer astronomy data also by reformatting low-level bit representations of values [58]. Instead of grouping bits by their significance, an *effective bit-width* is used alongside DEFLATE to achieve better compression than the combination of BitShuffle and LZ4 ($\mathcal{R}_c = 3.36$ vs $\mathcal{R}_c = 2.49$) on an integer dataset from the Five Hundred Meter Aperture Spherical Radio Telescope (FAST) [57].

2.1.3 Other Applications

Given their mapping abilities, SFCs have been used in numerous spatial and multidimensional applications other than compression. There are many such curves,

and many applications requiring these mappings; with different access requirements, computational behaviour, and parameters over which to optimize. The property of SFCs that is utilized in these scenarios is called *locality preservation*, the ability to keep nearby points close when mapping between one and d dimensions. Many of these applications involve improving the performance of data access in databases or storage. SFCs have been used to cluster multi-attribute objects together so that similar attribute values are grouped together, an important factor in the performance of database engines [3], [59], [60]. This clustering has also been used to reduce the I/O time to retrieve data from magnetic storage [59] as well as improving disk address prediction in certain contexts [61].

Common spatial data structures have benefited from the use of SFCs to map the d-dimensional data and generate optimized versions of the data structure itself [4], [10], [62]–[64]. This is particularly useful for k-Nearest Neighbour queries. These curves have even been used to improve the performance of machine-learning algorithms. Instances of this in the literature include the classification of schizophrenia and normal patients' 3D fMRI scans [65], gesture recognition of Surface Electromyography (sEMG) signals represented as images [66], [67], and malware detection and classification using Support Vector Machines [68].

2.2 Locality Preservation

Given a mapping f, there are numerous properties that can be identified. The mapping could be continuous, bijective, or differentiable. All of these properties deal classify functions and how they operate on input parameters. One such property is whether f is *order preserving*, or monotonic, in which case the following condition must be met.

$$f(x_i) \leq f(x_j), \quad \forall x_i, x_j \in \mathbb{X}, x_i \leq x_j \tag{2.12}$$

A *distance-preserving* mapping has the property defined in (2.13), where the distance measure, appropriate to the value-space, between values α and β is $d(\alpha, \beta)$. A *distance-preserving* transformation is an isometry [69].

$$d(x_i, x_j) = d(f(x_i), f(x_j)), \quad \forall x_i, x_j \in \mathbb{X} \tag{2.13}$$

Order and distance-preservation deal with the relationship between point-pairs for a given mapping. They are conditions that must be met for a mapping f to be classified as such. However, known SFCs are neither order nor distance preserving. Therefore, the concept of locality preservation is needed to quantify how well these mappings translate between one and d dimensions. Though it is not a well-defined term, locality preservation deals with the extent to which a mapping maintains the neighbourhood of points: points that are close by in the one-dimensional space are also close by in the d-dimensional space. There are multiple interpretations of this property, leading to different metrics and measures to quantify locality preservation. This introduces the possibility for conflicting definitions, but allows one to interpret *locality* within the context of its application. The work presented here principally deals with these interpretations and the research on their properties. However, there are certain mappings which are not constrained to a singular number of dimensions. Cantor shows that the cardinality of the set of all real numbers $|\mathbb{R}|$ is the same as the d-dimensional set of all real numbers $|\mathbb{R}^d|$ [24]. This implies that a mapping $f\colon \mathbb{R} \mapsto \mathbb{R}^d$ may be bijective, but Netto shows that it cannot be bijective and continuous [38]. In this case, the mapping from one-dimension to d-dimensions cannot be an isometry as that would imply it is both bijective and continuous. Order-preserving and distance-preserving deal with the relationship between point-pairs and a mapping. However, locality-preserving deals with neighbouring points and the degree to which that neighbourhood is preserved. This may appear to be a generalization of the distance-preservation properties, but it is more flexible in its definition, allowing for interpretation by the reader and those applying it to a specific use-case. This is evident in the discussion and analysis of metrics quantifying locality preservation in Chapter 3.

2.3 Metrics, Measurements, and Metric Types

Metrics are quantitative representations of properties from certain systems or models that allow for their ranking and analysis. Without metrics, it is difficult to compare multiple parameters as either they are different quantities and thus a comparison is not well-defined, or there are too many to practically make such a comparison. Therefore, parameters and measurements on the system are typically aggregated into fewer representative values called metrics. In this work metrics quantifying locality preservation are compared not only in how they rank SFCs (the system or model), but also how they are related to each other based on their definitions. To carry out this analysis, it is important to define the methodology by which to evaluate all metrics covered here.

A metric M is a method by which measurements m_i on a system are aggregated into a singular value. For a given SFC $\mathcal{S}$, there may be multiple measurements $m_i(\mathcal{S})$, but for a given metric, there is only one metric value $M(\mathcal{S})$. Each measurement extracts information about underlying properties of the SFC. By aggregating over all or a subset of the measurements, the metric can be tuned to represent certain general properties of the SFC. For the metrics covered here, the choice of which measurements to include impacts how well the metric ranks these curves. Applying this methodology to the example of a computational experiment, the system is a combination of the software being tested, the algorithms, and the hardware on which the experiments are executed. Measurements made on this system may include the execution time and the memory usage of the software for a single run of its program. One may choose to conduct these measurements on a subset of possible input parameters for the software and algorithm, thus constraining the scope of the final metric. With a certain number of measurements and experiments completed, an average or maximal metric may be taken. Instead of comparing all measurements, the aggregation process summarizes all measurements into a singular value suitable for comparison. This process is shown below, where agg is an aggregating operator (such as max or avg).

$$M(\mathcal{S}) = \underset{i}{\text{agg}} \{m_i(\mathcal{S})\} \tag{2.14}$$

In some instances, one cannot necessarily find an exact analytical representation for a given metric and SFC, as is the case with many of the metrics discussed in this work. However, it is sometimes possible to identify bounds on the metric values, showing that a metric M falls within a specified range for a given SFC. The metric may never reach these limits, which is why so much research is conducted on identifying tighter bounds for popular metrics and SFCs. Using the aforementioned example, a possible lower bound could be that the memory usage is always above a minimum required to launch the program. The program may never reach this lower-bound as the algorithm could consume a non-zero quantity of memory. Note that these bounds are proven to be correct, in that the metric can never exceed them. Similar upper bounds are used in the analysis of algorithms, with the worst-case and average time-complexities.

For the rest of this document the notation for metrics and measurements used is as shown in this subsection. Upper-case symbols denote metrics M, which are aggregates over measurements denoted using the lower-case of the same symbol m. In cases where metrics and measurements inherit from a generalized type, they are parameterized by superscript and subscript notation.

There are multiple metrics quantifying some form of locality preservation, with different applications giving rise to different measurements. They can be categorized into two types: point-pair metrics and region metrics. The former deals with measurements on pairs of points along the curves. The latter deals with measurements on points encompassed by a given d-dimensional region, such as a hyperrectangle. Both types focus on different properties of SFCs and are ways of quantifying how appropriate these curves are for different applications. These applications are discussed further in Sections 3.1, 3.2 and 3.4. Within each type there are multiple subtypes, parameterized to allow for the *extraction* of different properties of the curves. One such parameter is p, which denotes the type of d-dimensional distance used: the p-norm. Given that this is foundational to one's understanding of metrics,

the definition of the p-norm and some notation for subsequent metrics are given here.

Following from the notation in Section 2.1, the *distance* between two indices x_i and x_j is equivalent to the absolute difference between their values, the index-difference: $|x_i - x_j|$. The distance between two d-dimensional points is measured using the p-norm $\|K_i - K_j\|_p$ and is defined as the pth root of the sum of the coordinates to the power of p, where p is a positive real number. This is given below in (2.15) where $K_i = (\kappa_{i,0}, \ldots, \kappa_{i,d-1})$.

$$\|K_i - K_j\|_p = \left(\sum_{l=0}^{d} \{(\kappa_{i,l} - \kappa_{j,l})^p\} \right)^{\frac{1}{p}} \tag{2.15}$$

The above definition is somewhat different for the infinite norm,

$$\begin{aligned} \|K_i - K_j\|_\infty &= \lim_{p' \to \infty} \left(\sum_{l=0}^{d} \left\{ (\kappa_{i,l} - \kappa_{j,l})^{p'} \right\} \right)^{\frac{1}{p'}} \\ &= \max_l \{(\kappa_{i,l} - \kappa_{j,l}\}. \end{aligned} \tag{2.16}$$

To maintain the same nomenclature for the p-normed distance throughout this work, we abuse the notation and say that the infinite norm has $p = \infty$. For the purpose of computation, points are treated as d-dimensional vectors containing coordinate values. Special cases of the p-norm are where $p \in \{1, 2, \infty\}$: $\|\cdot\|_1$ is the Manhattan Distance, $\|\cdot\|_2$ is the Euclidean Distance, and $\|\cdot\|_\infty$ is the Infinity Norm or Chebyshev Distance (equivalent to the maximum component). In some literature, these are indicated with L-1, L-2, and L∞ respectively. However, some metrics utilize L as the metric symbol, resulting in conflicting notation. In this work, unambiguous notation is used to avoid confusion to the reader. Notation used in the literature is included where it would not cause ambiguity or where it is useful to the reader.

2.4 Datasets

Two separate datasets were identified as appropriate for evaluating the impact of SFC reordering on compression schemes: Digital Elevation Model (DEM) data and

radio-astronomy correlator data. Their properties are summarized in Table 2.1. The former is a selection of files from the Shuttle Radio Topographical Mission (SRTM) sourced through the United States Geological Survey (USGS). The dataset files have a spatial resolution of 1-arc second or 30 metres per pixel [70], [71]. When Memon, Neuhoff, and Shende conclude that the Hilbert Curve does not improve prediction gains for context-based predictive lossless encoders, they do so under the assumption that the data can be modelled as an Isotropic Gaussian Random Field (IGRF) [54]. Existing literature indicates that DEM data can be simulated using IGRFs [16], [72], making the SRTM dataset a good real-world candidate for the model chosen by Memon, Neuhoff, and Shende [54]. Empirical evidence for the SRTM data following this model is given in Section 2.4.

The SRTM dataset is significantly larger than the three files used by Scarmana and McDougall [12] and the hundred files used by Rane and Sapiro [55]: three 8 MB files versus one hundred 2.75 MiB files versus nine hundred and twenty 8 MiB files. Their data and the SRTM dataset have the same data type (signed 16 bit integers) though only that used by Scarmana and McDougall and the SRTM dataset have similar extents per file (2000×2000 versus 2048×2048 pixels). The DEM files used by Rane and Sapiro measure 1201×1201 pixels in size.

The second dataset is sourced from the South African Radio Astronomy Observatory (SARAO) archive [73], which is the South African component of the Square-Kilometre Array (SKA) radio-astronomy project [19]. Once completed, the SKA will be the largest radio-telescope in history. Originally created in 2018, this specific SKA dataset stores electromagnetic interference measurements in an HDF5 file [21]. The SKA measures the electromagnetic power-spectrum from hundreds of dishes, positioned around both Southern Africa and Australia, which is then correlated and processed to filter out noise and amplify low-power emissions from distant sources. The resulting data are three-dimensional in nature, with components *time*, *channel*, and *correlator products* [74]. Each channel is a fixed band of the frequency spectrum while the correlator products encode the measurements' distribution throughout the physical array. The data are stored as IEEE-754 standard single-precision floating-point values [75], presenting different bit-level behaviour to the SRTM integer data. Though typically two floating-point values are stored, representing the complex

power-spectrum, the specific dataset acquired has large regions where only the real power-spectrum is present. Other datasets from the SKA and SARAO contain the full complex power-spectrum though permission could not be acquired to utilize them. For this reason, the imaginary component was removed to not bias the results presented in this research. As is discussed in Section 2.4, the SKA dataset does not present the same locality correlation and isotropic behaviour as the SRTM files, which is one of the reasons the dataset was chosen. By comparing datasets with different data types and models, the impact of the SFCs on compression performance can be investigated in more detail.

Data Preparation

The SRTM and SKA datasets were sourced in the GeoTIFF [76] and HDF5 [21] formats respectively. To maintain a common format from which to apply the SFCs and compression techniques, a custom binary file-format called SFCC was created to store the same values as those stored in the original dataset files. Each SFCC file contains a nine-byte header storing metadata to assist subsequent processing steps. This includes the number of bytes in the data type, number of dimensions, the sidelength of the data extents, the current SFC ordering applied, the current

Table 2.1: Properties of the Two Investigated Datasets. *Data Type* Names are from the *C/C++* Language Standards in Little-Endian. The File Properties Shown Here are for the Processed and Prepared Files Acquired from the Original Datasets. The Names and Labels for Each Dimension in the Given Datasets are Provided.

	Processed Datasets	
	SRTM (A)	SKA (B)
Total Size	7.72 GB	7.28 GB
File Size	8 388 617 B	8 388 617 B
Data Type	*int16*	*float32*
File Shape	2048×2048	$128 \times 128 \times 128$
# Files	920	868
Dim. 1	Longitude	Time
Dim. 2	Latitude	Channel (frequency)
Dim. 3	—	Correlator Products

compression scheme applied, whether BitShuffle is applied or not, and a four-byte magic-word to identify SFCC files. Each dataset was split into standalone files which maintained the same dimensionality as the original data but with different file extents. The sidelengths were kept as powers-of-two, given the mathematical constraints on the chosen SFCs, such that files from both datasets have the same total file-size of 8.38 MB.

Each SRTM GeoTIFF file was divided into four overlapping quadrants to translate the 3601×3601 data extents into four 2048×2048 SFCC files (extents is measured in pixels or values). The SKA data were chunked into cubes of sidelength 128 with no overlap. Owing to the data types being two and four bytes for the SRTM and SKA datasets respectively, the resulting SFCC files all measure just over 8.38 MB. The final processed datasets contain 868 SKA and 920 SRTM SFCC files, where the values are stored in row-major order.

Dataset Models

The autocorrelation data plotted in Fig. 2.2 support the pre-existing assumptions that the SRTM data, being a DEM dataset, can be modelled as an Isotropic Gaussian Random Field. The X and Y plots in Fig. 2.2 show that elevation-values are highly correlated when they are close in the d-dimensional space. The similarity between X and Y indicates that the SRTM data are not biased to either dimension and can be classified as isotropic. The SKA data show different characteristics, with a sharper reduction in the autocorrelation, more so given the smaller sidelength used (128 versus 2048). The discrepancies between the *time*, *channel*, and *correlator* dimensions indicate that the SKA data may be anisotropic, though the lack of significantly high correlation means this conclusion cannot be confirmed in its entirety. However, it is important to note that unlike the SRTM data, the SKA data reach a noticeable floor of zero correlation for the *correlator* dimension, indicating a finite range of non-zero correlation in its direction. The bias towards the *time* dimension is hypothesized to originate from EMI sources measured by the instruments, where the electromagnetic spectrum channels used are fixed and the duration of interference is sustained, creating characteristic *FFT waterfall* patterns in the data [77].

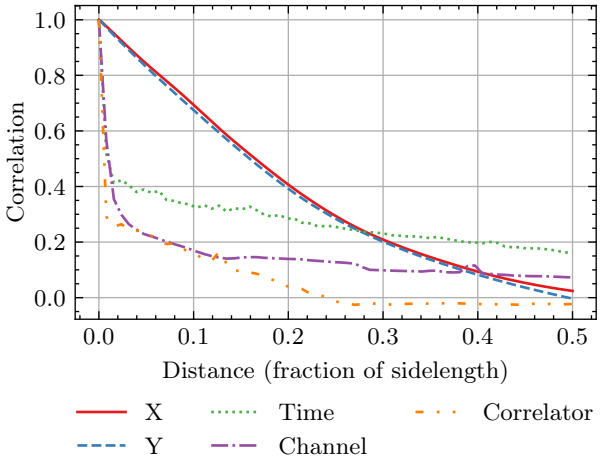


Figure 2.2: Autocorrelation for both datasets along all dimensions. The distance is measured as a fraction of the d-dimensional SFC-space sidelength N. X and Y are longitude and latitude in the SRTM data. Time, Channel, and Correlator are the three dimensions in the SKA data. Note that the sidelength for the SKA data is a 16th of that of the SRTM data: 128 versus 2048.

2.5 Compression Schemes

Compression schemes can be split into multiple categories, with each individual scheme potentially being comprised of multiple preprocessing steps and compression algorithms. As an example, BZIP2 is a combination of Run-Length Encoding (RLE), the Burrows-Wheeler Transform (BWT), Move-to-Front Encoding (MTF), and Huffman Encoding (HFF) [78], [79]. Compression algorithms are typically defined by the general technique used to reduce the size of dataset. Entropy Encoding, Dictionary Compression, Arithmetic Compression, and Predictive Coding are a few examples of such general techniques. Table 2.2 marks which preprocessing techniques and compression algorithms make up the compression schemes and data-formats identified in the aforementioned literature and sources of the chosen datasets. HDF5 is not listed as a standalone format and instead compatible compression schemes are marked with a footnote marker.

Four standalone algorithms were chosen as good candidates for this work as they are used as components of many of the schemes and formats identified: Huffman

Table 2.2: Composition of Compression Schemes and File-Formats Split into Preprocessing Steps (Preproc.) and Compression Algorithms. Arithmetic Coding (ARITH.) and Predictive Coding (PRED.) are Truncated to Reduce Space Requirements. Schemes have their own Software and Tools that can Compress Files Whereas Algorithms are Standalone Methods for Compressing. The DEFLATE Algorithm is Indicated by $\times^1$. Instances where the Implementation of a Given Algorithm Differs from the Canonical Definition are Marked with $\times^2$. Schemes[3] Denote Compression Tools that are Supported by HDF5. Algorithms and Schemes Investigated are Underlined.

	Preproc.			**Compression Algorithms**						
Schemes	BWT	MTF	ST	ARITH.	HFF	JPEG	LZ77	LZW	PRED.	RLE
BZIP2[3]	×	×			×					
GZIP[3]					$\times^1$		$\times^1$			
LZ4[3]							$\times^2$			
LZO[3]							$\times^2$			
SZIP[3]			×	×						
Unix Comp.								×		
ZSTD[3]					$\times^1$		$\times^1$			
Formats										
GeoTIFF					$\times^2$	×		×		$\times^2$
IMG										$\times^2$
JPEG-LS									×	
PNG					$\times^1$		$\times^1$			
SRTM DTED										
USGS DEM										

Encoding (HFF), LZ77, LZW, and Run-Length Encoding (RLE). BZIP2 and GZIP were included as they are commonly found in popular Linux distributions [80]. LZ4 [56] and LZO [81] were chosen as they are modifications of LZ77 that are supported by HDF5, the format for the SKA dataset.

The implementations of HFF [82], LZW [82], RLE [82], and LZ77 [83] differ from other implementations as their canonical definitions do not define file-format requirements and other implementation details. For example, the Huffman encoding software used employs canonical variable-length codes, a static tree, and stores the tree in the first 256 bytes of the compressed data-block. The HFF implementation used in GZIP/DEFLATE can use either fixed or dynamic codes [51]. As is discussed in

Section 4.2, the implementation of RLE used is somewhat *naive* as runs of 1 are not encoded differently to longer runs, thus increasing file sizes where run lengths are short and/or infrequent. Technical parameters for the compression methods used are given in Table 2.3.

Table 2.3: Configuration Parameters for Compression Schemes Used. Block Size Refers to the Size of Independently Compressed Sections of Data. Symbol Size is the Number of Bytes from the Data that are Treated as a Standalone Value for Dictionary Creation and Encoding Purposes. Some Values are not Listed as they are Implied by the Compression Level of the Given Tool. [1]The RLE Symbol Size Shown is for *int16* and *float32* Data Respectively. [2]Some Values are the Maximum Sizes to which that the Metric is Allowed to Expand.

Comp. Type	Block Size	Dictionary Size	Symbol Size (bytes)	Comp. Level
BZIP2	900 kB			9
GZIP			1	9
HFF			1	
LZ4	4 MB			9
LZ77		64 KiB[2]	1	
LZO	256 KiB			9
LZW		6 KiB[2]	1	
RLE			2,4 bytes[1]	

Of the eight schemes used in this research, half are provided by *external* tools within the Linux testing environment. The *internal* schemes are integrated directly into the compression testing software as they are canonical algorithms without a standard tool or program. Preliminary testing did not significantly compress the SKA dataset. To explore the lack of compressibility in these data and enhance compression performance, the BitShuffle process is implored as an extra preprocessing step for both datasets and all compression schemes.

2.5.1 BitShuffle

BitShuffle is a bit-wise preprocessing technique designed to enhance the subsequent compression of radio-astronomy data, specifically for HDF5 [20]. It extends the foundational premise of shuffle, a preprocessing filter integrated into HDF5, where bytes are reordered into groups by their significance in their higher-level data type [21], [84]. BitShuffle reorders the data by bit instead of byte and, when combined with lossy precision reduction and LZ4, has been shown to achieve a compression ratio of about 3.58 on radio data [20]. In the published definition of BitShuffle, Masui *et al.* found the next best compression ratio at 3.30 with DEFLATE at a compression level of seven but with an order-of-magnitude slower speeds [20]. Of the other compression schemes they cover, the one with speeds closest to the aforementioned BitShuffle combination achieves a compression ratio around 40 % worse. As this work only covers lossless techniques, the variant of BitShuffle used here does not employ the precision reduction or LZ4 compression step covered by Masui *et al.* [20]. Instead, here and for the rest of this document, BitShuffle refers to only the preprocessing bit-level restructuring. Applying BitShuffle to dataset $\mathbb{D}$,

$$\mathbb{D} = \{v^0, v^1, v^2, \ldots, v^N\},$$

where v^i has the binary structure

$$v^i = v_0^i \; v_1^i \; v_2^i \; \cdots \; v_b^i,$$

results in the BitShuffled output

$$\mathbb{D} \overset{BitShuffle}{\rightarrow} \{\bar{v}^0, \bar{v}^1, \bar{v}^2 \ldots, \bar{v}^b\},$$

where $\bar{v}^i$ has the binary structure

$$\bar{v}^i = v_i^0 \; v_i^1 \; v_i^2 \; \cdots \; v_i^N.$$

If N is divisible by eight, then $\bar{v}^i$ is byte-aligned. The BitShuffle process does not increase the file-size. The metadata indicating that BitShuffle has been applied to an SFCC file is encoded in the most significant bit of the header data type byte.

2.5.2 Compression Notation

Given that a dataset $\mathbb{D}$ is most likely stored in a one-dimensional representation, it must first be transformed into the d-dimensional space for the subsequent SFC mapping $\mathcal{S}$ to be applied. As most multidimensional data are in either row or column-major order, the Raster Scan SFC $\mathcal{R}$ is used: $\mathcal{R}(\mathbb{D}) = \mathbb{D}^d$. By reordering a given dataset $\mathbb{D}^d$ using an SFC, $\mathcal{S}^{-1}\left(\mathbb{D}^d\right)$, the locality of the points is somewhat preserved: i.e. neighbouring points in the d-dimensional space are *close* in the one-dimensional space. Note that the inverse of the SFC mapping $\mathcal{S}$ is taken as the reordering must output a one-dimensional array. The full process, from row-major order to the SFC mapped array is defined as $\mathcal{S}^{-1}(\mathbb{D}^d) = \mathcal{S}^{-1}(\mathcal{R}(\mathbb{D}))$.

Part of the contribution of this work is in the evaluation of how effectively this reordering improves the compression performance of some compression techniques. The initial hypothesis was, given the evidence in the literature, that a compression scheme $\mathcal{C}$ would reduce the total size of a dataset more if an SFC mapping is used. The hypothesis is shown in (2.17) where $|\mathbb{D}|$ denotes the total size of the dataset $\mathbb{D}$ and $\mathcal{C}(\mathbb{D})$ is the application of compression technique $\mathcal{C} \in \mathbb{C}$ to the same dataset.

$$\left|\mathcal{C}\left(\mathcal{S}^{-1}\left(\mathbb{D}^d\right)\right)\right| < \left|\mathcal{C}\left(\mathcal{R}^{-1}\left(\mathbb{D}^d\right)\right)\right| \tag{2.17}$$

The compression ratio $\mathcal{R}_c$ is defined as the ratio between the uncompressed dataset and the compressed dataset as shown in (2.18), with larger ratios indicating better compression. As is discussed in Section 4.2, the extent to which the compression performance is improved is heavily dependent on the dataset and whether BitShuffle is applied.

$$\mathcal{R}_c = \frac{|\mathbb{D}|}{|\mathcal{C}(\mathbb{D})|} \tag{2.18}$$

Chapter 3

A Review of Metrics Quantifying Locality Preservation of Space-Filling Curves and their Applications

The paper covering the investigation, comparison, and analysis of metrics quantifying locality preservation of Space-Filling Curves, to be submitted to the IEEE Transactions on Computational Imaging, is given here. Within this chapter metrics in the literature are identified and categorized into types and subtypes. Variations on the type parameters are discussed, generalizing the metrics as much as possible. The metric values for different SFCs are analysed, identifying which curves are *best*. Finally, an analysis is given into how the metrics and measurements relate to the operations used in certain applications. In this analysis, metrics appropriate for scientific and parallel computing, and data compression are identified; applications in line with the aim and objectives of this dissertation. The measurement-aggregate-measurement methodology in Section 2.3 is used to conduct this analysis. This paper and chapter form one of the contributions of this work; addressing the first research question and objective for this research in Sections 1.1 and 1.3 respectively. The current reference for this paper, until accepted by the peer-review process, is

> **C. J. Haupt**, E. J. Otoo, and L. Cheng, "A review of metrics quantifying locality preservation of Space-Filling Curves and their applications," To be Submitted to the IEEE Transactions on Computational Imaging, 2021.

Sections relevant to this paper, whose content is included in the paper manuscript, are Sections 2.1 to 2.3. The core of the research in this paper was conducted by **C.J. Haupt**, under the guidance and supervision of Prof. E.J. Otoo and Prof. L. Cheng. The associated analysis and discussion was conducted by **C.J. Haupt**. The paper

was written by **C.J. Haupt** under the supervision of Prof. L. Cheng, ensuring the quality and standard necessary for international publication was achieved.

3.1 Point-Pair Metrics

Point-Pair metrics quantify locality preservation by making measurements on points K_i and K_j in either or both of the SFC spaces: one and d dimensions. In certain cases, such as with Hölder Continuity Ratios, the ratio of the distance between the points in both SFC spaces is taken. No matter how the point-pairs are used, there are two components to all of these metrics: which points pairs are selected and how the measurements are aggregated. The following subsections cover four Point-Pair metrics, where the first three are closely related and, in certain cases, can be said to be measuring the same quantities. These are the Hölder Continuity Ratio, Reciprocal Hölder Continuity Ratio, and Partition Ratio. These metrics are variants of each other and have been referred to as *locality ratios* in the literature [2]. However, they can easily be separated into three distinct categories, as is done here.

3.1.1 Hölder Continuity Ratio

The Hölder Continuity, or Hölder Condition, is used to analyse functions in d dimensions, their derivatives, and to what extent they are continuous. A function $f(x) \in \mathbb{R}^d, \forall x \in \mathbb{R}$ is Hölder Continuous with exponent q if and only if the following Hölder Condition is satisfied [85]:

$$\sup_{\substack{x_1,x_2\in\mathbb{R} \\ x_1\neq x_2}} \frac{\|f(x_1) - f(x_2)\|_2}{|f(x_1) - f(x_2)|^q} \leq +\infty \tag{3.1}$$

The Hölder Continuity can be further refined, as is done by Bader, such that a finite non-zero real coefficient C can be used to describe an SFC $\mathcal{S}(x_i)$ [86]. This refactoring results in the following equation,

$$\frac{\|\mathcal{S}(x_i) - \mathcal{S}(x_j)\|}{|x_i - x_j|^q} \leq C, \tag{3.2}$$

for all unique pairings of curve indices x_i and x_j. The coefficient C is applicable for all point-pairs and is thus an upper-bound on what we call the Hölder Continuity Ratio (HCR), the left-hand-side of (3.2). Therefore, most metrics based on (3.2) treat the left-hand-side as the measurement over which they aggregate to rank a given curve. Of the form illustrated here, Gotsman and Lindenbaum were the first to publish a formal definition and analysis on its use [34], [87]. Their definition is given in (3.3) with two variants L_1 and L_2. The former is as given in the aforementioned equation while the latter is the minimum over all points instead of the maximum.

$$L_1(\mathcal{S}) = G_{\mathrm{GL}}^{\max}(\mathcal{S}) = \max_{\substack{x_i, x_j \in \mathbb{M}_x \\ x_i < x_j}} \left\{ \frac{\|\mathcal{S}^{-1}(x_i) - \mathcal{S}^{-1}(x_j)\|_2^d}{|x_i - x_j|} \right\} \tag{3.3}$$

Though Chochia, Cole, and Heywood applied the Manhattan Distance to (3.3) [88], it was Alber and Niedermeier that generalized the definition to any p-norm [89]. To denote such metrics and their parameters, we extend (3.2) and (3.3) further, utilizing similar definitions by Alber and Niedermeier [89]. The generalized HCR measurement is given in (3.4) where p, q, and r are positive non-zero real parameters that alter the behaviour of the measurements. Variants of the HCR in the literature, along with the parameter values used, are given in Table 3.1. Their subtypes are given in Fig. 3.1 where the literature are grouped by parameter values.

$$g_{p,q,r}(\mathcal{S}, x_i, x_j) = \frac{\|\mathcal{S}^{-1}(x_i) - \mathcal{S}^{-1}(x_j)\|_p^r}{|x_i - x_j|^q}. \tag{3.4}$$

Metrics aggregating over $g_{p,q,r}$ measurements typically include all unique point-pairs. We denote such metrics as $G_{p,q,r}$.

However, there are further constraints made in the literature which we can use to refine $G_{p,q,r}$. All twelve of the identified papers investigating the HCR define their metrics as the maximum of the $g_{p,q,r}$ measurements. The only exception is the work by Gotsman and Lindenbaum where they use the minimum and maximum, as described in (3.3). However, they do differ in which p-norm is taken. In combining their works into a succinct description for the HCR metric, we define $G_p^{\max}$ and

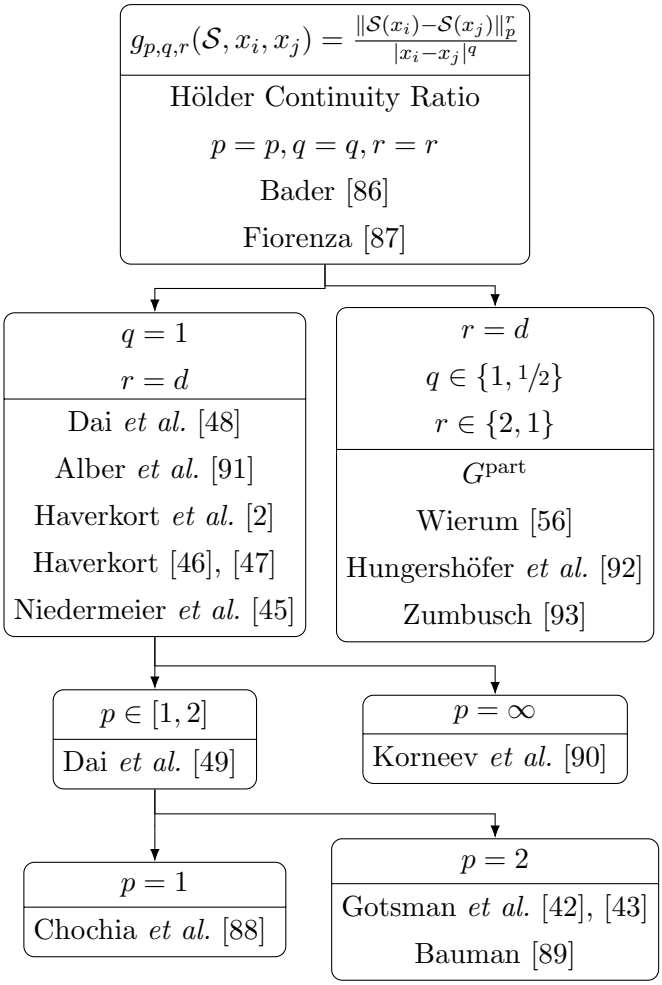


Figure 3.1: Hölder Continuity Ratio Subtype-Tree illustrating the *inheritance* of parameters and constraints within metrics from the literature. Parameters used are p, q, and r. Inheritance of constraints is explicitly for the operand g and not for the method of aggregating the point-pair values.

Table 3.1: Hölder Continuity Ratio Metrics, Their Associated Arguments, and Relevant Literature. Specifics of the Bounds on the Metric Values are also Given.

		Source	p	q	r	m	d	Bounds	Curves
HC	C	Bader [86]	$\{2, p\}$	q					
		Fiorenza [85]	p	q					
HCR	$G_p^{\max}$	Chochia, Cole, and Heywood [88]	1	1	2	≥ 2	2	exact	$\mathcal{H}$
		Gotsman and Lindenbaum [34], [87][b]	2	1	2	$m, \geq 6$	$\{2, 3\}$	upp/low	$\mathcal{S}, \mathcal{H}$
		Bauman [90]	2	1	2	∞	2	exact	$\mathcal{H}$
		Dai and Su [91]	$[1, 2]$	1	2	m	2	upp	$\mathcal{H}$
		Korneev and Shchepin [92]	$\{2, \infty\}$	1	3	m	3	exact	$\mathcal{H}$[a]
		Dai and Su [93]	$\{1, 2, \geq 2\}$	1	2	m_0	2	exact	$\mathcal{H}$
		Alber and Niedermeier [89]	$\{1, 2, \infty\}$	1	2	m	2	upp/low	$\mathcal{H}$
		Niedermeier, Reinhardt, and Sanders [32]	$\{1, 2, \infty\}$	1	2	m	2	upp/low	$\mathcal{H}, \mathsf{H}, \mathcal{S}$
		Haverkort and van Walderveen [2]	$\{1, 2, \infty\}$	1	2	∞	2	upp	$\mathcal{P}, \mathsf{H}, \mathcal{H}, \mathcal{Z}, \beta\Omega, \mathcal{A}$, More
		Haverkort [62], [94][d]	$\{1, 2, \infty\}$	1	3	∞	3	upp	$\mathcal{H}$ and variants
	G^{part}	Wierum [30][c,d]	2	$\{1, 1/2\}$	$\{2, 1\}$	m	2	upp	$\beta\Omega, \mathsf{H}, \mathcal{Z}, \mathcal{H}$
		Hungershöfer and Wierum [95][c]	2	$\{1, 1/2\}$	$\{2, 1\}$	m	2	upp/low	$\mathcal{H}, \mathcal{Z}$
		Zumbusch [96][d,e]	2	$\{1, 1/2\}$	$\{2, 1\}$	m	2	upp/low	$\mathcal{H}, \mathcal{Z}$

[a]The Tokarev Curve is used, a variant of the Hilbert Curve.
[b]Sources investigate both the minimum and maximum of the given metric.
[c]Sources investigate both the maximum and average of the given metric.
[d]Sources identify some bounds using empirical/computational results for the given metric.
[e]Sources investigate only the maximum of the given metric.
G^{part} refers to both G^{diam} and G^{surf}.
m_0 denotes an SFC-Order above an arbitrary order.

$G_p^{\min}$ as the maximum and minimum of $g_{p,1,d}$ respectively. These metrics fix the second and third parameters, $q = 1$ and $r = d$, to ensure that measurements do not diverge as the size of the SFCs increase, such as with the limit as their sidelengths N tends to infinity. If q and r are not chosen appropriately, the limit $\lim_{N\to\infty} g_{p,q,r}$ would diverge owing to incompatible orders for the numerator and denominator in (3.4): $g_{p,q,r}$ scales with order $\frac{O(N^r)}{O\left(N^{qd}\right)} = O\left(N^{r-qd}\right)$. To ensure this limit converges, the exponent $r - qd$ must be zero. All literature identified have $q = 1$ and $r = d$.

Bader highlights how the ratio can converge with $q = 1/d$ and $r = 1$, though none of the cited literature on the HCR use this method [86]. However, it is encountered with the Partition Ratio in Section 3.1.2. As $q = 1$ and $r = d$ are fixed, the only other tunable parameter is p, with $p \in \{1, 2, \infty\}$ being the most popular values. Six papers cover more than one value for p, with two of those investigating values other than 1, 2, and ∞, both by Dai and Su: $p \in \mathbb{R} \geq 2$ [93] and $p \in [1, 2]$ [91]. The other three papers are authored by Alber and Niedermeier [89], Niedermeier, Reinhardt, and Sanders [32], and Haverkort and van Walderveen [2].

3.1.2 Partition Ratio

The Partition Ratio (PR) metric resembles the Hölder Continuity Ratio in Section 3.1.1, but instead measures over ranges of indices called partitions P [30], [95], [96]. Where only indices x_i and x_j are used by the HCR, the PR investigates all indices bounded by them: $P(x_i, x_j) = \{x_l | x_i \leq x_l \leq x_j\}$. Given that the norm of a sequence of points is not well-defined, the PR replaces the numerator in (3.4) with the partition's diameter $\text{diam}(\mathcal{S}, x_i, x_j)$ and surface area $\text{surf}(\mathcal{S}, x_i, x_j)$. The diameter of a given partition is defined in (3.5), while the surface area is the d-dimensional surface area of the partition in the multidimensional space. Note that the d-dimensional surface area is equivalent to the perimeter in the two-dimensional case.

$$\text{diam}(\mathcal{S}, x_i, x_j) = \max_{x_i \leq x_1 < x_2 \leq x_j} \{\|K_1 - K_2\|_2\}. \tag{3.5}$$

The PR's measurements are the diameter $g^{\text{diam}}(\mathcal{S}, x_i, x_j)$ and surface $g^{\text{surf}}(\mathcal{S}, x_i, x_j)$ ratios for a given partition constructed by indices x_i and x_j. Their definitions are given in (3.6) and (3.7).

$$g^{\text{diam}}(\mathcal{S}, x_i, x_j) = \frac{\text{diam}(\mathcal{S}, x_i, x_j)^2}{|x_i - x_j|} \tag{3.6}$$

$$g^{\text{surf}}(\mathcal{S}, x_i, x_j) = \frac{\text{surf}(\mathcal{S}, x_i, x_j)}{4 \times \sqrt{|x_i - x_j|}} \tag{3.7}$$

The denominator has a different meaning in the context of partitions, having the same mathematical definition as in g but instead being the size or volume of the partition. There are three subtleties to the PR metric. Firstly, the diameter is measured as the Euclidean Norm and not parameterized. Secondly, equivalent parameters exist for r and q in (3.6) and (3.7). They serve the same purpose, of ensuring the measurements do not diverge as the sidelengths of the SFC-space increase. Given that the surface area of a partition scales differently to the diameter, the Partition Surface Ratio (PSR) requires different values for r and q to ensure convergence. The values $q = \frac{1}{2}$ and $r = 1$ are used, as well as a $\frac{1}{4}$ multiplier, as seen in (3.7) [96].

As is the case with many of the other metrics discussed in this paper, the average and maximum over all partitions is taken as the metric aggregating function. The Partition Diameter Ratio (PDR) metrics are given in (3.8) and (3.9) and the PSR metric are given in (3.10) and (3.11). The maximum PDR is the same as the maximum HCR in (3.3).

$$G^{\text{diam}}_{\text{avg}}(\mathcal{S}) = \underset{\substack{x_i, x_j \in \mathbb{M}_x \\ x_i < x_j}}{\text{avg}} \left\{ g^{\text{diam}}(\mathcal{S}, x_i, x_j) \right\} \tag{3.8}$$

$$G^{\text{diam}}_{\text{max}}(\mathcal{S}) = \underset{\substack{x_i, x_j \in \mathbb{M}_x \\ x_i < x_j}}{\max} \left\{ g^{\text{diam}}(\mathcal{S}, x_i, x_j) \right\} \tag{3.9}$$

$$G^{\text{surf}}_{\text{avg}}(\mathcal{S}) = \underset{\substack{x_i, x_j \in \mathbb{M}_x \\ x_i < x_j}}{\text{avg}} \left\{ g^{\text{surf}}(\mathcal{S}, x_i, x_j) \right\} \tag{3.10}$$

$$G^{\text{surf}}_{\text{max}}(\mathcal{S}) = \underset{\substack{x_i, x_j \in \mathbb{M}_x \\ x_i < x_j}}{\max} \left\{ g^{\text{surf}}(\mathcal{S}, x_i, x_j) \right\} \tag{3.11}$$

The PDR indicates the locality preservation of a given SFC whereas the PSR is a measure of the partition quality [30], [95], [96]. The PSR is also called the Quality Coefficient, with lower values being of a higher quality. This is because the such partitions are used in Finite Element Modelling analysis where the *quality* of the partitions is important [96]. An SFC with a low PSR is well suited for these parallel compute-intensive applications.

3.1.3 Reciprocal Hölder Continuity Ratios and EdgeSums

Before Gotsman and Lindenbaum published their analysis on the Hölder Continuity Ratio; Perez, Kamata, and Kawaguchi analysed the Hilbert curve using what is equivalent to the inverse of the Hölder Continuity Ratio [97], which we call the Reciprocal Hölder Continuity Ratio (RHCR). Their work quantifies locality preservation by defining a metric that penalizes neighbouring points, in d dimensions, that are far apart along the curve. They achieve this by stating that an ordering of coordinates, using an SFC, should minimize the weighted sum of one-dimensional distances, where the weights are inversely proportional to the d-dimensional distance [97]. Using the same parameters, and notation as with the HCR $g_{p,q,r}$, we define the Reciprocal Hölder Continuity Ratio as

$$f_{p,q,r}(\mathcal{S}, x_i, x_j) = \frac{|x_i - x_j|^q}{\|\mathcal{S}^{-1}(x_i) - \mathcal{S}^{-1}(x_j)\|_p^r}. \tag{3.12}$$

The weighted sum used by Perez, Kamata, and Kawaguchi, which is to be minimized, is given as

$$F_p^{\mathrm{PKK}}(\mathcal{S}) = \sum_{\substack{x_i, x_j \in \mathbb{M}_x \\ x_i < x_j}} \{f_{p,1,1}(\mathcal{S}, x_i, x_j)\}. \tag{3.13}$$

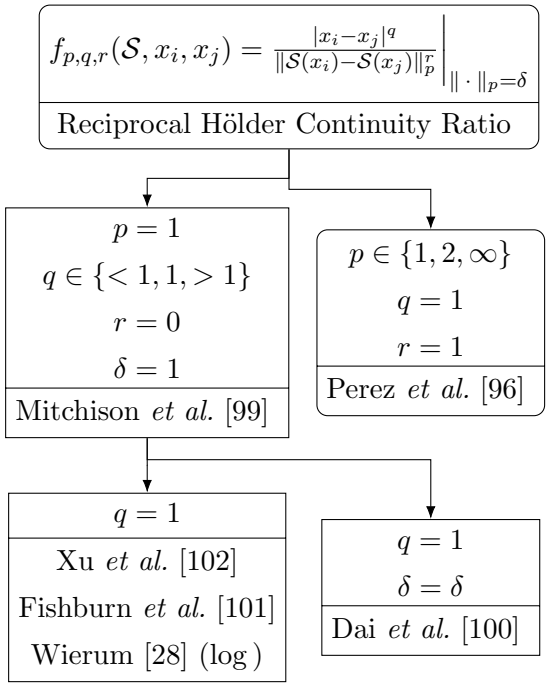


Figure 3.2: Reciprocal Hölder Continuity Ratio Subtype-Tree illustrating the *inheritance* of parameters and constraints within metrics from the literature. Parameters used are p, q, r, and δ. Inheritance of constraints is explicitly for the operand f and not for the method of aggregating the point-pair values. The middle-right node with Pérez, Benavent, and Olanda does not take into account δ. Square-edged nodes are also referred to as EdgeSum metrics.

Note that Perez, Kamata, and Kawaguchi do not utilize q and r to mitigate divergence at the limit as is the case with $g_{p,q,r}$. This is observed in the SFC performance discussed in Section 3.3. Literature that utilizes and investigates the RHCR are not immediately identifiable, as many instead cover the EdgeSum metric: an unweighted sum of edges [99], [100]. Details on this literature is collated in Table 3.2. The EdgeSum metric does not utilize the d-dimensional distance in the measurement as is the case with the RHCR. Instead, the d-dimensional distance is used to constrain which point-pairs over which to measure.

Table 3.2: Reciprocal Hölder Continuity Ratio Metrics, Their Associated Arguments, and Relevant Literature. Specifics of the Bounds on the Metric Values are also Given.

		Source	p	q	r	δ	m	d	Bounds	Curves
RHCR	$F_{p,q,r}$	Perez, Kamata, and Kawaguchi [97]	$\{1, 2, \infty\}$	1	1					$\mathcal{H}$
	$F_{q,\delta}^{\text{edge}}$	Dai and Su [101]	1	1	0	δ	m	$\{2, 3\}$	exact	$\mathcal{Z}, \mathcal{H}$
		Fishburn, Tetali, and Winkler [102]	1	1	0	1	m'	2	exact	$\mathcal{M}$
		Mitchison and Durbin [100]	1	1[a]	0	1	m'	2	low/exact	$\mathcal{M}$
	F^{nns}	Xu and Tirthapura [103]	1	1	0	1	m	d	low/exact	$\mathcal{S}, \mathcal{Z}, \mathcal{R}$
	$F^{\log}$	Wierum [104]	1	1	0	1	m	2	exact	$\mathcal{H}, \mathcal{Z}$

[a]Sources investigate $q < 1$, $q = 1$, and $q > 1$, but only $q = 1$ is shown in the table for formatting reasons. m' denotes any positive non-zero real-numbered SFC-order. The given metrics do not discriminate sidelengths that are not of the form $N = b^m$.

This can be seen in Fig. 3.2, where, for three of the metric tree nodes, r is set to zero. The EdgeSum metric is typically defined for graphs, and thus assumes explicit edges between points. For the RHCR-based EdgeSum used here, an edge between points K_i and K_j implicitly exists if and only if the following condition is satisfied:

$$\|K_i - K_j\|_p = \delta. \tag{3.14}$$

The weight of each edge is the numerator in the RCHR measurement equation, (3.12). Setting $\delta = 1$ implies that the d-dimensional points of an SFC form a rectilinear lattice with axis-aligned edges between neighbouring points. Of the five sources in the literature that investigate the EdgeSum metric, only one does not assume $\delta = 1$, by Dai and Su [101]. Of the same five sources, four assume $q = 1$ [101]–[104]. The fifth work covers both $q = 1$ and $q \geq 1$. The EdgeSum metric's measurement is given below.

$$f^{\text{edge}}_{q,\delta}(\mathcal{S}, x_i, x_j) = \begin{cases} f_{1,q,0}(\mathcal{S}, x_i, x_j) & \|K_i - K_j\|_1 = \delta \\ 0 & \text{otherwise} \end{cases} \tag{3.15}$$

Some of the EdgeSum metric definitions in the literature represent (3.15) differently, but they are equivalent in their application [100], [102]. Equation (3.15) is a restructured form of their definitions to match the HCR measurement equations. Similarly to the goal of minimizing $F^{\text{PKK}}_p(\mathcal{S})$, as stated by Perez, Kamata, and Kawaguchi, the EdgeSum metric is a summation over all valid measurements. A curve achieves *better* locality preservation the lower the value of this summation. This aggregation is given in (3.16). Definitions (3.15) and (3.16) can be restructured such that the summation limit only covers valid point-pairs, instead of setting the measurements $f^{\text{edge}}_{q,\delta}$ to zero if an edge *does not exist* between said points.

$$F^{\text{edge}}_{q,\delta}(\mathcal{S}) = \sum_{\substack{x_i, x_j \in \mathbb{M}_x \\ x_i < x_j}} \left\{ f^{\text{edge}}_{q,\delta}(\mathcal{S}, x_i, x_j) \right\}. \tag{3.16}$$

Two of the five EdgeSum metrics alter the definition, even though the same measurements are made. The first is by Wierum, called the Logarithmic EdgeSum [104].

As with $F_{q,1}^{\text{edge}}$, only neighbouring points construct valid point-pairs. The measurement $f^{\log}$ is defined as the logarithm of the edge distance, as shown in (3.17). The Logarithmic EdgeSum employs two aggregations, the maximum and average, shown in (3.18) and (3.19).

$$f^{\log}(x_i, x_j) = \begin{cases} \log_2(|x_i - x_j| + 1) & \|K_i - K_j\|_1 = 1 \\ 0 & otherwise \end{cases} \tag{3.17}$$

$$F_{\max}^{\log}(\mathcal{S}) = \max_{\substack{x_i, x_j \in \mathbb{M}_x \\ x_i \neq x_j}} \left\{ f^{\log}(x_i, x_j) \right\} \tag{3.18}$$

$$F_{\text{avg}}^{\log}(\mathcal{S}) = \operatorname*{avg}_{\substack{x_i, x_j \in \mathbb{M}_x \\ x_i \neq x_j}} \left\{ f^{\log}(x_i, x_j) \right\} \tag{3.19}$$

Wierum employs $f^{\log}(x_i, x_j)$ to determine how appropriate a grid ordering is for searching and sorting, such as with finite-element simulations [104]. As the number of operations for these queries is often described as $O(c \log d)$, it is logical to use the Logarithmic EdgeSum as an appropriate measurement for such a metric as it scales with the same order [104].

The second altered EdgeSum metric is utilized by Xu and Tirthapura [103]. Though it uses the same edge weights as the EdgeSum $f_{q,\delta}^{\text{edge}}$, it does not aggregate over the weights themselves but instead over the average and maximum weights for each point [103]. They call this metric the Nearest-Neighbour Stretch (NNS), as it indicates how much *stretch* there is from d-dimensional neighbouring points to their one-dimensional indices. The associated measurements are made on individual points and their respective neighbours. For a given point K_i, the neighbouring points are those connected to K_i by an implicit edge. The NNS measurement extends (3.15) by aggregating over these neighbours. The average-NNS and maximum-NNS measurements are given in (3.20) and (3.21) respectively.

$$f^{\text{nns}}_{\text{avg}}(\mathcal{S}, x_i) = \underset{\substack{x_i, x_j \in \mathbb{M}_x \\ \|K_i - K_j\|_1 = 1}}{\text{avg}} \{f_{1,1,0}(\mathcal{S}, x_i, x_j)\} \tag{3.20}$$

$$f^{\text{nns}}_{\text{max}}(\mathcal{S}, x_i) = \underset{\substack{x_i, x_j \in \mathbb{M}_x \\ \|K_i - K_j\|_1 = 1}}{\max} \{f_{1,1,0}(\mathcal{S}, x_i, x_j)\} \tag{3.21}$$

The NNS metrics aggregate these measurements over all points, giving the average-average NNS $F^{\text{nns}}_{\text{avg}}$ and average-maximum NNS $F^{\text{nns}}_{\text{max}}$ in (3.22) and (3.23) respectively. This differs from the EdgeSum metric $F^{\text{edge}}_{q,\delta}$ as individual edge weights may be *counted* twice, once for each point connected by a given edge. This is guaranteed for the average-average NNS but not necessarily for the average-maximum.

$$F^{\text{nns}}_{\text{avg}}(\mathcal{S}) = \underset{x_i \in \mathbb{M}_x}{\text{avg}} \left\{ f^{\text{nns}}_{\text{avg}}(\mathcal{S}, x_i) \right\} \tag{3.22}$$

$$F^{\text{nns}}_{\text{max}}(\mathcal{S}) = \underset{x_i \in \mathbb{M}_x}{\text{avg}} \{ f^{\text{nns}}_{\text{max}}(\mathcal{S}, x_i) \} \tag{3.23}$$

3.1.4 Description and Irregularity Vectors

Mokbel, Aref, and Kamel have published multiple works on analysing locality and order preservation quality of SFCs, called the Description Vector [36], [105] and Irregularity Vector [106], [107] respectively. Both vectors classify point-pairs into different types based on the kind of traversal between the points in each dimension. The Description Vector V deals only with point-pairs constructed from adjacent indices whereas the Irregularity Vector V_I encompasses all point-pairs. Each vector classifies pairs with respect to each dimension and then totals the number of pairs in each category, or segment type. There are four published papers on the Description and Irregularity Vectors, which are listed in Table 3.3 alongside the metric bounds identified by Mokbel, Aref, and Kamel. The Description Vector along dimension l contains five segment types; represented by the number of jump J_l, contiguity C_l, reverse R_l, forward F_l, and still S_l pairs. A specific point-pair may only fall in to a maximum of two of the segment types for a given dimension, such that the following equation holds true:

Table 3.3: Description and Irregularity Vector Metrics, Their Associated Arguments, and Relevant Literature. Specifics of the Bounds on the Metric Values are also Given.

Source	d	m	Bounds	Curves
Mokbel, Aref, and Kamel [36], [105]	2-12	≤ 8	exact	$\mathcal{R}, \mathcal{K}, \mathcal{Z}, \mathcal{G}, \mathcal{H}$
Mokbel and Aref [106], [107]	2-128	≤ 10	exact	$\mathcal{R}, \mathcal{K}, \mathcal{Z}, \mathcal{G}, \mathcal{H}$

$$J_l + C_l + S_l = R_l + F_l + S_l. \tag{3.24}$$

Given that a d-dimensional point K_i is composed of d coordinate values $(\kappa_{i,0}, \kappa_{i,1}, \ldots, \kappa_{i,d-1})$, the number of point-pairs under each specific segment type, along dimension $0 \leq l < d$ for a given SFC, are given as

$$J_l(\mathcal{S}, N, l) = \sum_{\substack{x_i, x_j \in \mathbb{M}_x \\ x_i+1=x_j}} \begin{cases} 1 & |\kappa_{i,l} - \kappa_{j,l}| > 1 \\ 0 & \text{otherwise} \end{cases}, \tag{3.25}$$

$$C_l(\mathcal{S}, N, l) = \sum_{\substack{x_i, x_j \in \mathbb{M}_x \\ x_i+1=x_j}} \begin{cases} 1 & |\kappa_{i,l} - \kappa_{j,l}| = 1 \\ 0 & \text{otherwise} \end{cases}, \tag{3.26}$$

$$R_l(\mathcal{S}, N, l) = \sum_{\substack{x_i, x_j \in \mathbb{M}_x \\ x_i+1=x_j}} \begin{cases} 1 & \kappa_{i,l} > \kappa_{j,l} \\ 0 & \text{otherwise} \end{cases}, \tag{3.27}$$

$$F_l(\mathcal{S}, N, l) = \sum_{\substack{x_i, x_j \in \mathbb{M}_x \\ x_i+1=x_j}} \begin{cases} 1 & \kappa_{i,l} < \kappa_{j,l} \\ 0 & \text{otherwise} \end{cases}, \tag{3.28}$$

$$\text{and } S_l(\mathcal{S}, N, l) = \sum_{\substack{x_i, x_j \in \mathbb{M}_x \\ x_i+1=x_j}} \begin{cases} 1 & \kappa_{i,l} = \kappa_{j,l} \\ 0 & \text{otherwise} \end{cases} \tag{3.29}$$

Mokbel, Aref, and Kamel further define the total counts for each segment type — J_T, C_T, R_T, F_T, and S_T — as the sum over all dimensions. The Description Vector for a given dimension l is $V_l(\mathcal{S}) = (J_l, C_l, R_l, F_l, S_l)$ and the Total Description

Vector is $V_T(\mathcal{S}) = (J_T, C_T, R_T, F_T, S_T)$. For continuous SFCs, such as the Hilbert and $\beta\Omega$ curves, the number of jump segments will always be zero; and non-zero for non-continuous SFCs, such as the Z-Order curve.

The Irregularity Vector is similar to the Description Vector in that it groups point-pairs into types; however, it is computed over all point pairs, not just those with adjacent indices. Furthermore, there is only one type: an irregularity [106], [107]. An irregularity exists, in dimension l, where the point corresponding to the largest index in the point-pair has a smaller coordinate value for the given dimension. The number of irregularities for a given dimension is defined as:

$$I_l(\mathcal{S}, N, l) = \sum_{\substack{x_i, x_j \in \mathbb{M}_x \\ x_i < x_j}} \begin{cases} 1 & \kappa_{j,l} < \kappa_{i,l} \\ 0 & \text{otherwise.} \end{cases} \tag{3.30}$$

The Total Irregularity Vector V_I contains the number of irregularities along each dimension — $V_I = (I_0, \ldots, I_{d-1})$ — while the Total Irregularity I_T is the sum of the irregularities in each dimension. The Irregularity and Description Vectors allow the analysis of SFC orderings within the context of *dimensional bias*: to what extent a given SFC favours one dimension over the other. The opposite is fairness, where the number of each segment type across all dimensions is similar. Given that V_I and V deal with the dimensional ordering of point-pairs as opposed to their locality, they are more suited to quantifying order-preservation. However, the ability to compare SFCs based on their fairness can be argued to be necessary for locality preservation. Therefore, the Irregularity and Description Vectors play a key part in evaluating locality preservation of a given d-dimensional ordering. Understanding dimensional-bias is important as some multidimensional SFC applications, such as with database indexing, require one dimension to take priority over the other. For real-time data, queries are typically bounded first by time, indicating that an SFC that biases towards one dimension may be better suited for such applications [107].

3.2 Region Metrics

Region metrics quantify locality preservation but measuring how tightly an SFC fills a given region. This differs from Point-Pair metrics in that points are grouped together based on their position within the d-dimensional space. There are two metrics covered in this section: Bounding Box metrics and the Clustering metric. The former can be said to be both a Point-Pair and Region metric as it operates on regions constructed from point-pairs. However, the measurements are made on the regions themselves and is thus categorized as a Region metric in this review. The Clustering metric measures the number of consecutive runs of indices contained within predefined regions, with each run being a *cluster*. The regions, types of measurements on said regions, and other parameters for both metrics and their associated literature are given in Table 3.4.

3.2.1 Bounding Region Metric

Many spatial data structures, such as R-Trees, use regions to group the underlying values. Haverkort and van Walderveen quantify how appropriate SFCs are for ordering these data structures by defining bounding regions around point-pairs along a given curve [2]. These bounding regions are representative of the regions used in spatial data structures, and improve query performance the smaller their surface area and volume, as this reduces the number of intersecting regions for a given query. Haverkort and van Walderveen investigate the two-dimensional case [2] whereas Haverkort later extends this to the three-dimensional case [62], [94].

For a given point-pair K_i and K_j, the associated bounding region is the *smallest* region of a given shape such that the partition between x_i and x_j is enclosed by said region. If between the two points, an SFC traverses far from the local context, then the bounding-region would be large as it must include all points within the partition. There are three types of regions covered by Haverkort and van Walderveen: the bounding-box bbox, bounding-octagon boct, and p-normed bounding-ball bball_p. A bounding-box is the same as a hyperrectangle and a bounding-ball is the same as a p-normed n-sphere or hypersphere. A bounding-ball of size r_s encompasses

Table 3.4: Region-Based Metrics, Their Associated Arguments, and Relevant Literature. Specifics of the Bounds on the Metric Values are also Given.

		Source	Region	Measurement	d	Bounds	Curves
BBox	*WBA*	Haverkort and van Walderveen [2][a]	Rect./Oct.	Surf./Vol.	2	upp	$\mathcal{P}, \mathsf{H}, \mathcal{H}, \mathcal{Z}, \beta\Omega, \mathcal{A}$, More
		Haverkort [62], [94][a]	Rect./Circ.	Diam./Surf./Vol.	3	upp	$\mathcal{H}$ and variants
Clusters	$H_{\max}$	Asano *et al.* [31]	Sqr.	Number of Clusters	2	upp/exact	RSFC, $\mathcal{A}$
		Moon *et al.* [35][a]	Sqr./Rect./Circ.		$\{2,3\}$	exact	$\mathcal{Z}, \mathcal{H}, \mathcal{G}$
	H_{avg}	Xu and Tirthapura [40]	Rect.[b]		2	low/exact	$\mathcal{S}, \mathsf{S}$
		Faloutsos and Roseman [11][a]	Sqr.		$\{2,3,4\}$	exact	$\mathcal{Z}, \mathcal{H}$
		Jagadish [59][a]	Line/Rect.		2	exact	$\mathcal{R}, \mathcal{K}, \mathcal{Z}, \mathcal{G}, \mathcal{H}$
		Jagadish [108]	Sqr.		2	exact	$\mathcal{H}$
		Xu, Nguyen, and Tirthapura [33][a,d]	Sqr./Rect.		$\{2,3\}$	exact	$\mathcal{H}, \mathcal{O}$

[a]Some bounds were identified using empirical/computational results.
[b]Regions include axis-aligned rotations; either all or a subset of all possible rotations.
[c]The given metric is defined for both the maximum and average of the measurements.
[d]The distribution of the given metric measurements is investigated.
Moon *et al.* [35] explore both $H_{\max}$ and H_{avg}.

all points at a distance, from the centre K_i, of at most $\|K_i - K_j\|_p$. Therefore, a bounding-ball using the Euclidean Norm is the same as a canonical hypersphere. The surface area and volume of these regions are the measurements used by the Bounding-Region metric. Note that the two-dimensional *surface area* and *volume* are equivalent to the perimeter and surface area respectively. In fact, Haverkort refer to the two measurements as the perimeter and area in their work from 2010 [2], but later change to the surface area and volume in 2011 [62]. To be consistent with other definitions of area and volume in this review, we use the later terminology.

Haverkort and van Walderveen define the Bounding-Region metrics as both the worst-case/maximum and average of the given measurements [2]. The Worst-Case Bounding-Box Surface Area *WBS* and Volume *WBV* Ratios are defined below, where $|\text{bbox}|$ denotes the volume of a given region and $\text{surf}(\text{bbox})$ its surface area. In the two-dimensional case, Haverkort define the Worst-Case Bounding-Box Surface Area *WBA* and Perimeter *WBP* Ratios. Though the previous two metrics could be used for $d = 2$, their definitions differ slightly for the two-dimensional [2] and three-dimensional [62], [94] cases. The same metrics for different regions require replacing bbox in the numerator with their associated bounding-region notation, as well as changing the acronym on the left-hand-side of Eqs. (3.31) to (3.34). All Bounding-Region metrics are taken at the limit $\lim_{N\to\infty}$, which is why the definitions below employ the limit superior.

$$WBS(\mathcal{S}) = \lim_{m\to\infty} \sup_{\substack{x_i,x_j\in\mathbb{M}_x \\ x_i<x_j}} \left\{ \frac{\text{surf}(\text{bbox}(K_i,K_j))^{\frac{d}{d-1}}}{2d^{\frac{d}{d-1}} \times |x_i - x_j|} \right\} \tag{3.31}$$

$$WBV(\mathcal{S}) = \lim_{m\to\infty} \sup_{\substack{x_i,x_j\in\mathbb{M}_x \\ x_i<x_j}} \left\{ \frac{|\text{bbox}(K_i,K_j)|}{|x_i - x_j|} \right\} \tag{3.32}$$

$$WBP(\mathcal{S}) = \frac{1}{16} \cdot \lim_{m\to\infty} \sup_{\substack{x_i,x_j\in\mathbb{M}_x \\ x_i<x_j}} \left\{ \frac{\text{per}(\text{bbox}(K_i,K_j))^2}{|x_i - x_j|} \right\} \tag{3.33}$$

$$WBA(\mathcal{S}) = \lim_{m\to\infty} \sup_{\substack{x_i,x_j\in\mathbb{M}_x \\ x_i<x_j}} \left\{ \frac{|\text{bbox}(K_i, K_j)|}{|x_i - x_j|} \right\} \tag{3.34}$$

The average metrics are not simply the average over all point-pairs for the given measurements. Haverkort and van Walderveen suggest that the average of the above metrics be the average over a uniformly sampled set of partitions [2]. However, they find that the number of samples impacts the resulting average metric in a periodic manner. Therefore, the average of the above metrics is the average, for different numbers of samples, of the average Bounding-Region measurements. These average metrics are only defined for the two-dimensional case (*ABA* and *ABP* [2]) as they are "much more difficult to compute efficiently and accurately for large numbers of curves" [94, p. 52]. However, Haverkort and van Walderveen does state that the average-case of the metrics may be more relevant in practice than the worst-case [2].

Haverkort also investigates other metrics which are equivalent to those described in Section 3.1 [62], [94]. They refer to the Hölder Continuity Ratio as the L_i-dilation or Worst-case WL_p, the Partition Diameter Ratio as the L_i-diameter ratio or WD_p, and the Partition Surface Ratio as the Surface Ratio or WS. In fact, the Worst-Case Bounding-Box Surface Ratio WBS is the Partition Surface Ratio where the partition is replaced with its bounding box. Technically, there can be as many variants of the Bounding-Region metric as there are region shapes and measurements on the bounding regions. An example of this is how Haverkort and van Walderveen investigate the Worst-case Bounding-Octagon Area *WOA* in two dimensions, which is the same as the Worst-case Bounding-Box Area in two dimensions *WBA*, Worst-case Bounding-Ball Volume in three dimensions WBB_p, and Worst-case Bounding-Box Volume in three dimensions WBV; other than a change of region shape. The two consistent components of the Bounding-Region metrics is the usage of the volume and surface area of the bounding regions. Haverkort and van Walderveen utilize these measurements as their metrics are intended to reduce the number of *buckets* retrieved for a range query on a bounding-region hierarchical data structure, such as an R-Tree [2].

3.2.2 Clustering Metric

The Clustering metric quantifies the number of continuous runs of indices within a given region. Initially this was to predict the I/O requirements for a given d-dimensional range query, as data organized using an SFC would require $\eta - 1$ hard-disk seeks for η clusters [11], [59]. In some cases, the SFC is used to traverse a database where each dimension is a different field or index [59], [108], [109]. The measurements are made for a region R of a given size r_s. In the literature, as shown in Table 3.4, there are four region types that are investigated: square $\square$, rectangular $\square$, circular $\bigcirc$, and line regions. Square and circular regions can easily be defined using a single region-size parameter r_s, being the sidelength and diameter respectively. In this section, as well as Section 3.3, the region-size parameter r_s is the maximum sidelength of a rectangular region, where all other sidelengths are a fraction of r_s. A line region is equivalent to a partial-match query, where some coordinates or fields are already known [59]. This is done to improve the readability of the sections. In practice, some literature identifies exact formulae based on all sidelengths of a rectangular region [40].

The measurement equation used by the clustering metric is defined as the number of clusters $\eta(\mathcal{S}, R_i)$ for a given region shape R, position, and SFC $\mathcal{S}$. A unique position of a region R in the SFC space is notated as $R_i \in R$. Of the seven sources investigating the Clustering metric, two aggregate with the maximum and six with the average. One source investigates both the Maximum and Average Clustering metric [35]. These metrics are defined as

$$H_{\max}(\mathcal{S}, R) = \max_{R_i \in R} \{\eta(\mathcal{S}, R_i)\} \tag{3.35}$$

and

$$H_{\text{avg}}(\mathcal{S}, R) = \operatorname*{avg}_{R_i \in R} \{\eta(\mathcal{S}, R_i)\}. \tag{3.36}$$

Note that the set of all regions, for a given region shape, is denoted as R whereas a unique position for the region within the SFC space is denoted as R_i. The number of regions in the set depends on the SFC space and the region size. Xu and Tirthapura

extend this further by considering axis-aligned rotations of the regions [40]. The rotation λ of a region R is equivalent to the permutation of the regions sidelengths as the rotation λ maintains the region's axis-aligned properties, at least for the rectangular $\square$ and line regions. The square $\square$ and circular $\bigcirc$ regions are symmetrical and thus rotations do not affect their Clustering metric. Xu and Tirthapura investigate the clustering metric under the constraints of all possible rotations $\lambda \in \Lambda^*$ and a subset of rotations $\lambda \in \Lambda \subset \Lambda^*$ [40].

Xu, Nguyen, and Tirthapura also extend the Clustering metric, by normalizing the measurements over the hypothetical optimal Clustering for the given SFC-space [33]. In this scenario, the resulting measurements are not the number of clusters, but a scaling factor indicating how many more clusters are encountered by an SFC compared to the optimal. However, much of their contribution is in identifying bounds on the same clustering metric in (3.36).

Asano *et al.* investigate the Clustering metric as defined here, for square regions; however, they add one assumption to their analysis: that the number of clusters for a given region allows for the access of a constant multiple of extra regions [31]. This implies that a square region $\square$ of size $|\square| = r_s^2$ can include Cr_s^2 points. Asano *et al.* include this assumption as their application of the Clustering metric is to disk access patterns, where it is fast to continue reading data than seek to a new position if the extra data read is sufficiently short.

3.3 Metric Performance for SFCs

Much research has been conducted into the performance of specific SFCs for the aforementioned metrics. In many cases, this involves analytical bounds on a metric M for a given SFC $\mathcal{S}$. In some cases, metric values are found empirically, utilizing computational methods to calculate the metric within specific conditions. This section shows some results from the literature covering metrics from Sections 3.1 and 3.2. Values for the metrics fall under three types: lower, exact, and upper bounds. The first and last define a range outside which the metric for a specific SFC cannot fall. Exact bounds are the *exact* analytical equations for the metric, sometimes under

certain conditions. The types of bounds for each source in the literature is already listed in Tables 3.1 to 3.4. However, in this section the bounds are given for specific SFCs and sources. For a given metric M and SFC $\mathcal{S}$, the bounds are as follows:

$$\begin{aligned} M_{\text{low}} \leq M_{\text{exact}} \leq M_{\text{upp}} \\ \text{where} \quad M_{\text{exact}} = M(\mathcal{S}) \end{aligned} \tag{3.37}$$

Bounds that are found empirically are denoted as such, indicating that the metric may differ somewhat in situations outside the parameters used in the computation. In most of the cases where empirical results are used, these parameters are appropriate to make a conclusion about the metric and SFC for all practical purposes. However, some cases, as with Faloutsos and Roseman in Section 3.3.1 [11], the empirical results are too limited.

Given the vast variety of SFCs, only a few key curves are discussed in this section. They include the Z-Order, Hilbert, AR^2W^2 , Mitchison, $\beta\Omega$, H-Order, and Onion Curves, as well as generic and recursive SFCs. Where the metric values are plotted and the lines are horizontal (i.e. constant vs the sidelength N or region-size parameter r_s), the source in the literature identifies this as a bound at the limit as the parameter tends to the maximum possible: infinity for N and N for r_s.

One conclusion from this section is that many of the best or optimal SFCs are *bespoke*, having been *designed* for a given metric. Examples are the H-Order [32], $\beta\Omega$ [30], Mitchison [100], [102], AR^2W^2 [31], and Balanced Peano [2] Curves.

3.3.1 Hölder Continuity Ratio and Partition Ratio Metrics

Bounds identified in the literature for the Hölder Continuity Ratio (HCR) and the Partition Ratio (PR) metrics are shown in Fig. 3.3. These metrics have very similar definitions, with the maximum Partition Diameter Ratio (PDR) $G_{\text{max}}^{\text{diam}}$ and maximum HCR $G_{\text{GL}}^{\text{max}}$ being the same [30], [62]. The Partition Surface Ratio (PSR) is included here given its relationship to the PDR.

Gotsman and Lindenbaum show that the upper bound of the Hilbert Curve for $G_2^{\max}$ is much higher than the lower bound for generic SFCs [34]. They conclude that there must be a better SFC for $G_2^{\max}$, given the large difference. Niedermeier, Reinhardt, and Sanders [32] utilize work by Alber and Niedermeier [89] to define H-Indexing, or the H-Order Curve, an SFC based on the Sierpiński-Knopp curve [1], with better $G_p^{\max}$ locality than the Hilbert curve with $p \in \{1, 2, \infty\}$ [32]. In fact, Haverkort and van Walderveen show, in their extensive work from 2010, that the H-Order Curve has the best $G_p^{\max}$ locality amongst twelve other SFCs [2]. They conjecture that the H-Order Curve's metric values of $G_1^{\max} = 8$ and $G_2^{\max} = G_\infty^{\max} = 4$ are optimal for two dimensions [2]. Haverkort identifies metric values for three-dimension dimensional Hilbert-like SFCs [62]. Korneev and Shchepin identify bounds on $G_\infty^{\max}$ for the Hilbert curve in three dimensions, but not for $p \neq \infty$ [92]. Work by Wierum on the PR led to the development of the $\beta\Omega$ curve, which they show has a maximum PDR better than the Hilbert Curve [30], but the empirical work by Haverkort and van Walderveen shows that the H-Order Curve is better than both [2]. In comparing HCR metric bounds, Wierum shows that the $\beta\Omega$ curve lies between the Hilbert and Z-Order curves for the maximum PSR metric $G_{\max}^{\text{surf}}$ but beats them both for the average PDR metric $G_{\text{avg}}^{\text{diam}}$. However, the $\beta\Omega$ Curve has better average PSR than the Hilbert and Z-Order Curves [30].

Utilizing Fig. 3.3 as a guide for analysing the HCR and PR metrics, one can see that *worse* curves can achieve better (lower) metric values for small sidelengths. Obviously, these conclusions are made given the information available, and are not guarantees that a better curve may be identified or that better bounds are found in the future. For $p = 1$ and $N < 2^4$, the Hilbert Curve appears to be better than the H-Order Curve. However, for $N > 2^4$, the H-Order Curve is the best. The bounds for the $\beta\Omega$ and H-Order Curves are at the limit, and therefore a comparison with the Hilbert Curve for finite sidelengths cannot be made with only this data. Some plots of $G_1^{\max}$ for the Hilbert Curve have exact equations [88], [93]. In fact, Dai and Su show that the exact bound for $G_1^{\max}$ [93] is the same as the lower bound identified by Niedermeier, Reinhardt, and Sanders [32]. What is clear is that for large SFCs ($N >> 2^4$), the H-Order Curve achieves better $G_1^{\max}$ values than the Hilbert Curve. Similar conclusions can be made for $G_3^{\max}$ in two dimensions. However, the $\beta\Omega$ Curve does outperform the Hilbert Curve for large sidelengths [2], [30].

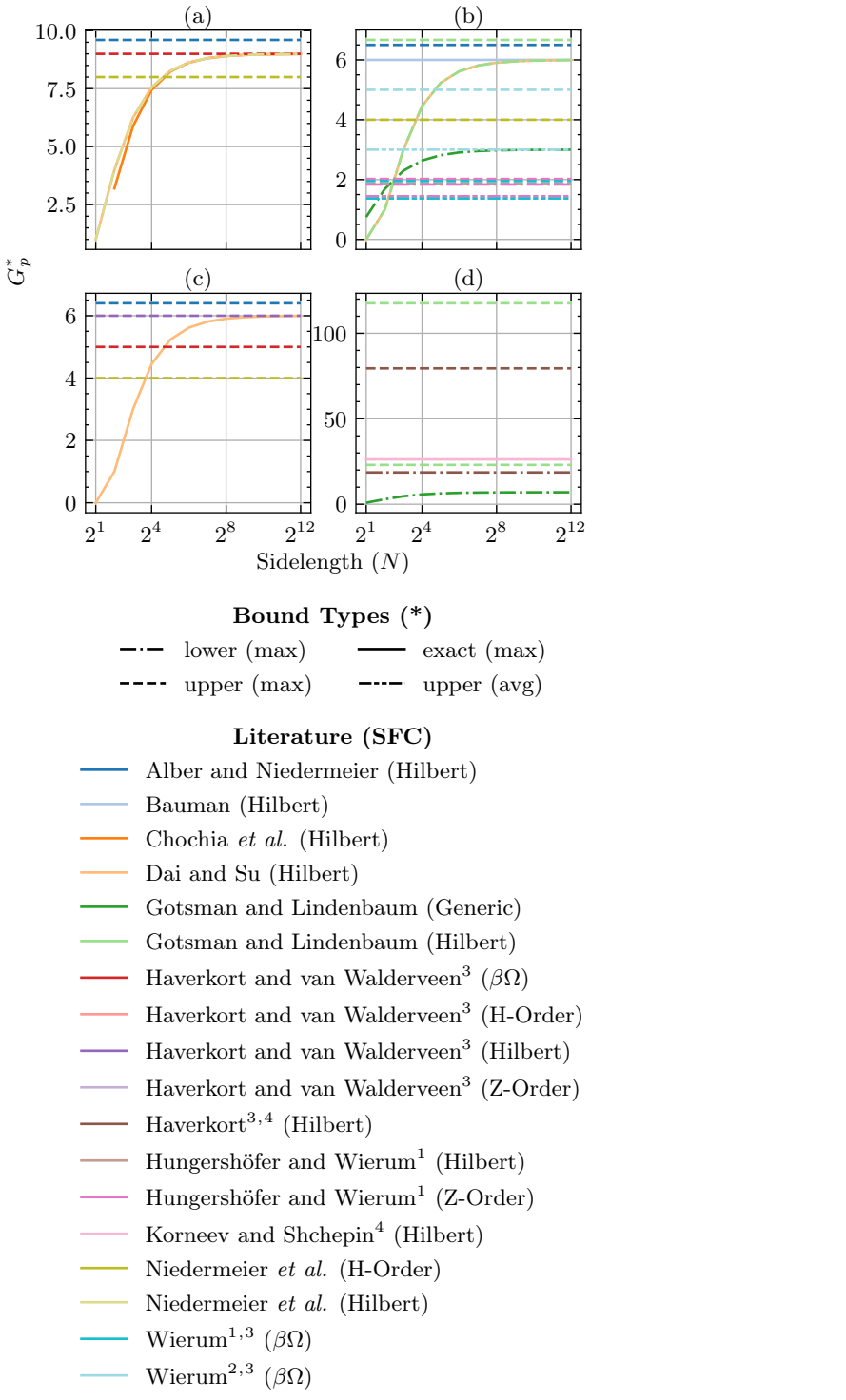


Figure 3.3: Some bounds for the Hölder Continuity Ratio and Partition Ratio metrics. The subfigures are for different combinations of parameters: (a) $p = 1, d = 2$, (b) $p = 2, d = 2$, (c) $p = 3, d = 2$, and (d) $p = 2, d = 3$. The exact plot for Dai and Su (Hilbert) is drawn underneath the lower-bound plot for Niedermeier *et al.* (Hilbert) in subfigure (a) and the lower-bound plot for Gotsman and Lindenbaum (Hilbert) in subfigure (b). [1]Plots are for the Partition Surface Ratio metric. [2]Plots are for the Partition Diameter Ratio metric. [3]Bounds are hypothetical as they were found empirically. [4]Bounds are for the Tokarev Curve, a variant of the Hilbert Curve.

There are many more sources investigating $G_2^{\max}$ than for $p = 1$ and $p = \infty$, in two dimensions. The maximum and average PSR values for the Hilbert, Z-Order, and $\beta\Omega$ Curves are between 1.4 and 2.1 [30], [95]. Similarly to $G_1^{\max}$ with the lower bound by Niedermeier, Reinhardt, and Sanders [32], Dai and Su show that the lower bound for the Hilbert Curve [34] is in fact the exact metric value [93]. Comparing the Hilbert Curve to the H-Order and $\beta\Omega$ Curves, as with the previous p values, the Hilbert Curve is the worst of the three for $N < 2^5$ as the exact and lower bounds for $G_2^{\max}$ [87], [93] exceeds the upper bounds on $G_2^{\max}$ for the H-Order [32] and $\beta\Omega$ [30] Curves.

Bounds from the literature, for the HCR and PR in three dimensions in three dimensions, are only identified for the Hilbert Curve and its variants, as seen in Fig. 3.3. Korneev and Shchepin identify the exact bound for the Tokarev Curve, an SFC based on the Hilbert Curve [92]. However, this variant of the Hilbert Curve has different $G_p^{\max}$ values compared to other Hilbert Curves. Haverkort, Haverkort identifies bounds for all Hilbert-like SFCs in three dimensions: $2.65^3 \leq G_2^{\max} \leq 4.3^3$. Note that the upper bound on $G_2^{\max}$ for the canonical Hilbert Curve is 23 [87], significantly lower than the upper bound for all Hilbert-like Curves found by Haverkort [62], [94].

Though some results shown are empirical [2], [30], [62], [94], analytical results from other literature [32], [34], [87], [88], [93] confirms the hypothesis that the conclusions made from the computational results appear accurate, even at the limit. This is also the case for analytical results at the limit [90] compared to analytical results as a function of the sidelength [34], [87], [93].

3.3.2 Reciprocal Hölder Continuity Ratio Metrics

Bounds identified in the literature for the Reciprocal Hölder Continuity Ratio (RHCR), Nearest-Neighbour Stretch (NNS), and EdgeSum metrics are shown in Fig. 3.4. Perez, Kamata, and Kawaguchi defines the RHCR, which utilizes the d-dimensional distance between points to normalize the metric measurements $f_{p,q,r}$ [97]. However, as discussed in Section 3.1.3, most of the literature either utilizes the

HCR or an EdgeSum metric, where $r = 0$. Therefore, none of the bounds plotted in Fig. 3.4 are for the RHCR F_p^{PKK}. Instead, the EdgeSum $F_{q,\delta}^{\text{edge}}$ [100]–[102], NNS $F_{\text{avg}}^{\text{nns}}$ and $F_{\text{max}}^{\text{nns}}$ [103], and Logarithmic EdgeSum $F_{\text{avg}}^{\log}$ and $F_{\text{max}}^{\log}$ [104] metrics are shown.

Though they do not refer to their ordering as a curve, Mitchison and Durbin identify an SFC that is optimal for the EdgeSum metric in a square SFC space [100]. Fishburn, Tetali, and Winkler independently identify the same curve, though they generalize it to a rectangular SFC space [102]. Fishburn, Tetali, and Winkler and Mitchison and Durbin obtain the same analytical equation for their bounds in Fig. 3.4(b), except for an extra term generalized as $O\big(N^2\big)$.

In the work presented here, this curve is called the Mitchison Curve $\mathcal{M}$. The specific variant of the curve shown in Fig. 2.1(g) is optimal for $q = 1$, with the curve having to be modified to be optimal for $q < 1$ and $q > 1$ [100]. Mitchison and Durbin identify a lower bound for the EdgeSum and Mitchison Curve [100], with Fishburn, Tetali, and Winkler finding that the exact EdgeSum metric matches the lower bound closely [102]. The optimality of the Mitchison Curve is seen in Fig. 3.4(b), where the Hilbert and Z-Order Curves have higher $F_{q,\delta}^{\text{edge}}$ metric values. Dai and Su find that the Z-Order Curve outperforms the Hilbert Curve for $F_{1,\delta}^{\text{edge}}$ where δ is an integral power-of-two [101]. They further define this by stating that the ratio between the metric values is approximately constant. For $\delta << N$ the Z-Order is superior with the EdgeSum metric,

$$\frac{F_{1,\delta}^{\text{edge}}(\mathcal{H}_m^d)}{F_{1,\delta}^{\text{edge}}(\mathcal{Z}_m^d)} \approx \begin{cases} 1.2143 & d = 2 \\ 1.0806 & d = 3, \end{cases} \tag{3.38}$$

but loses to the Hilbert Curve if the edge distance δ is set to the sidelength ($\delta = N$),

$$\frac{F_{1,\delta}^{\text{edge}}(\mathcal{H}_m^d)}{F_{1,\delta}^{\text{edge}}(\mathcal{Z}_m^d)} \approx 0.9682. \tag{3.39}$$

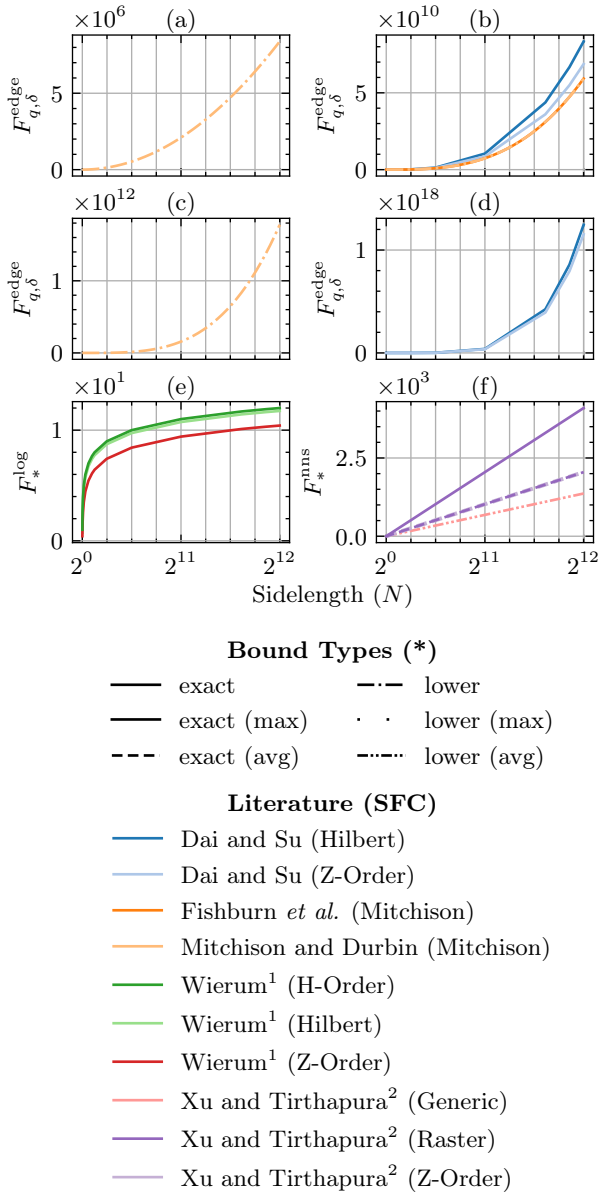


Figure 3.4: Some bounds for the EdgeSum, Nearest-Neighbour Stretch, and Logarithmic EdgeSum metrics. The subfigures are for different combinations of parameters: (a) $q < 1, d = 2$, (b) $q = 1, d = 2$, (c) $q > 1, d = 2$, (d) $q = 1, d = 3$, (e) $q = 1, d = 2$, and (f) $q = 1, d = 2$. Subfigures (e) and (f) are for the Logarithmic EdgeSum and Nearest-Neighbour Stretch metrics respectively. [1]Literature covers the Logarithmic EdgeSum. [2]Literature covers the Nearest-Neighbour Stretch. Bound Types without any parenthesis are for the EdgeSum. The aggregation denoted in the parenthesis is referenced in the notation by F_*.

All plots in Fig. 3.4 are with $\delta = 1$. In Section 3.1.3 it is stated that the EdgeSum and NNS metrics do not normalize in the same way as the Hölder Continuity Ratio, utilizing q and r to ensure the numerator and denominators scale with the same order. This statement is verified in Fig. 3.4 where $F_{q,\delta}^{\text{edge}}$ diverges as the sidelength increases. This divergence is also present for the Logarithmic EdgeSum $f_*^{\log}$ and NNS F_*^{nns}. However, they do not diverge in an exponential or polynomial manner. The Logarithmic EdgeSum scales, as is obvious, logarithmically. The NNS metric scales linearly, owing to the double aggregation process used: $F_{\text{avg}}^{\text{nns}}$ is the average of the average NNS measurement $f_{\text{avg}}^{\text{nns}}$.

Unfortunately, none of the literature calculates the Logarithmic EdgeSum or NNS for the Mitchison Curve. However, as with the EdgeSum metric, the Z-Order Curve outperforms the Hilbert Curve for $F_{\text{max}}^{\log}$ [104]. A more interesting result for the Logarithmic EdgeSum is that the H-Order and $\beta\Omega$ Curves are worse than the Hilbert Curve, and have the same exact maximum Logarithmic EdgeSum $F_{\text{max}}^{\log} = \log(N)$ [30]. However, the *best* SFC is different for the average Logarithmic EdgeSum $F_{\text{avg}}^{\log}$, with the $\beta\Omega$ Curve being the best and the Z-Order Curve being the worst [104]. The H-Order Curve is not as good in the average case compared to the maximum, being worse than the Hilbert Curve for the average Logarithmic EdgeSum metric. Values for the average Logarithmic EdgeSum metric found by Wierum are shown below [30].

$$F_{\text{avg}}^{\log}(\mathcal{Z}_m^2) = 2.7455 \pm 0.0005 \tag{3.40}$$

$$F_{\text{avg}}^{\log}(\mathsf{H}_m^2) \approx 2.58 \tag{3.41}$$

$$F_{\text{avg}}^{\log}(\mathcal{H}_m^2) = 2.533 \pm 0.018 \tag{3.42}$$

$$F_{\text{avg}}^{\log}(\beta\Omega_m^2) = 2.514 \pm 0.011 \tag{3.43}$$

Bounds on the Nearest-Neighbour Stretch (NNS), identified by Xu and Tirthapura, show that the Raster and Z-Order Curves have the same average-average NNS $F_{\text{avg}}^{\text{nns}}$ [103]. However, these bounds are at the limit as the sidelength tends to infinity. More specifically, the analytical equations for the average-average NNS found by Xu and Tirthapura is

$$F_{\text{avg}}^{\text{nns}}(\mathcal{H}_m^d) \sim \frac{1}{d} N_T^{1-\frac{1}{d}} \tag{3.44}$$

and

$$F_{\text{avg}}^{\text{nns}}(\mathcal{Z}_m^d) \sim \frac{1}{d} N_T^{1-\frac{1}{d}} \tag{3.45}$$

where

$$\begin{aligned} &f(x) \sim g(x) \\ &\text{if} \quad \lim_{x\to\infty} \frac{f(x)}{g(x)} = 1. \end{aligned} \tag{3.46}$$

Therefore, the NNS plots in Fig. 3.4 are not necessarily the real bounds for the given sidelength. The lower bound for $F_{\text{avg}}^{\text{nns}}$, with generic SFCs, is also a lower bound for the average-maximum NNS $F_{\text{max}}^{\text{nns}}$ [103]. This is a trivial lower bound as the maximum of a function will always be greater than or equal to the average. Xu and Tirthapura do not calculate the NNS for the Hilbert Curve [103]. However, Deford and Kalyanaraman compute the average NNS for the Hilbert and Gray-Code Curves, finding that the latter is the worst and the former is slightly worse than the Z-Order and Raster Curves [110]. They extend the Nearest-Neighbour Stretch metric to any Manhattan Distance δ, implying that larger areas form part of the measurements $f_{\text{avg}}^{\text{nns}}$ and $f_{\text{max}}^{\text{nns}}$. With their extended version of the metric, they show that the Hilbert and Z-Order Curves are better than the Gray-Code and Raster Curves. Therefore, one can conclude that the Z-Order and Raster Curves are beneficial for very close neighbouring points, as opposed to those further away; at least with the NNS metric. From the bounds shown here, it appears that the Z-Order and Mitchison Curves are the best, depending on the specifics of the RHCR-based metric used.

3.3.3 Irregularity and Description Vectors

Mokbel, Aref, and Kamel investigate the Raster, Snake, Z-Order, Gray-Code, and Hilbert Curves for the Description Vector [36], [105] and Irregularity Vector metrics [106], [107]. There are three methods with which they analyse the metrics: Scalability, Fairness, and Intentional Bias. The first pertains to how the percentage contribution of each vector element changes as a function of the number of dimensions d and the

SFC sidelength N. Fairness and Intentional Bias are opposite methods, with fair SFCs having no intentional bias. Fairness is measured as the standard deviation of a given vector segment type (such as a jump segment or irregularities in a given dimension) over all dimensions. If the standard deviation is low, then all dimensions are treated similarly by the SFC, and thus it is fair. Intentional Bias is also related to how the vector metric values vary between dimensions. However, the focus is on having all dimensions behave similarly, with one dimension containing significantly higher metric values.

Description Vector

For all curves investigated, Mokbel, Aref, and Kamel find that the percentage of still segments increases with the number of dimensions [36], [105]. However, the proportion of each segment type remains relatively constant as a function of the SFC sidelength N. The Gray-Code, Hilbert, and Raster Curves have similar scaling for the Reverse R, Forward F, and Still S segments; though the Gray-Code Curve has slightly more Jumps J and fewer Contiguity C segments. The Peano Curve has the highest number of Contiguity and Reverse segments and the lowest number of Still segments. Therefore, choice of a curve based on scalability is dependent on what segment type must scale better.

The Hilbert Curve is the fairest of the curves, with each segment type having low standard deviation for all number of dimensions covered by Mokbel, Aref, and Kamel [36]. However, they do note that the standard deviation for the Hilbert Curve does increase slightly as a function of d while the Peano and Gray-Code curves' decrease. Therefore, it may be that for $d >> 12$, the Hilbert Curve is not the fairest.

Given that the Hilbert Curve is the fairest, it cannot be that with the *best* intentional bias. As the Raster and Snake Curves fill an entire dimension, or row, before traversing to another; they have the highest intentional bias of the SFCs investigated by Mokbel, Aref, and Kamel [36]. The Gray-Code and Peano Curves have some intentional bias, definitely more so than the Hilbert Curve. The Gray-Code Curve has more intentional bias than the Peano Curve.

Irregularity Vector

The Raster and Peano Curves have the same scalability as a function of d for the Irregularity Vector metric, with the Gray-Code and Hilbert Curves having similar scalability [106], [107]. The former two have the best scalability, at least for $d < 64$, above which all SFCs achieve similar scaling. As a function of the SFC sidelength, the Raster and Peano Curves still achieve the best scaling. However, for $N > 128$ only the Raster, Peano, and Snake Curves converge to similar irregularity. The Gray-Code and Hilbert Curves always have worse scaling as a function of the SFC sidelength. Therefore, for $d < 64$, the best RSFC for scaling is the Peano Curve.

The Peano also beats the Gray-Code and Hilbert Curves in fairness, for $d < 10$. For more than ten dimensions, the latter two curves are fairer than the Peano Curve. The Gray-Code and Hilbert Curves maintain 50 % irregularity in all dimensions except the first. This impacts the fairness as a function of d as the impact of the first dimension is reduced as the number of dimensions increases. As a function of the sidelength, the fairness does not change above $N = 128$ similarly to the scalability. However, the *best* SFC does not change as a function of N, only d.

Any bias to a given dimension for the SFCs and the Irregularity Vector metric is constant above $d = 10$. The Raster and Snake Curves have the greatest bias, as all dimensions have an irregularity of 0 %, except for one with 50 %. Of the RSFCs, the Peano Curve has the best intentional bias as the least and most irregular dimensions have 32 % and 50 % irregularities respectively, a difference greater than the Gray-Code or Hilbert Curves'.

The Description and Irregularity Vector metrics have different *best* SFCs, as is indicated in this section. For example, the fairest SFCs for two metrics are the Hilbert and Peano Curves respectively. This is not to say that these two curves are not *good* for the other metric, only that they are the best. As can be expected, non-recursive SFCs are good at biasing towards a single dimension as they are not restricted to subquadrants within the d-dimensional SFC space. The reason the Peano Curve is the fairest and most biased for the Irregularity Vector has to do with how it scales: it is fairest for low dimensions and small sidelengths, but becomes the

most biased for high dimensions and large sidelengths. The Hilbert Curve fill the gaps where the Peano Curve is insufficient, being the fairest for high dimensions and large sidelengths. This is excluding non-recursive SFCs. For high dimensions and large sidelengths, all RSFCs achieve the same intentional bias.

3.3.4 Bounding-Region Metrics

Given the large number of subtypes for the Bounding-Region metric — being the combinations of two and three dimensions; box, octagon, and ball regions; and volume and surface area measurements — only a few important types and SFCs are discussed here [2], [62], [94]. The Worst-case and Average-case Bounding-Box Surface Area and Volumes are important for predicting the performance of hierarchical bounding-box data structures, like R-Trees. Computed metric values, for two-dimensional SFCs already discussed in this work, are given in Table 3.5 [2].

Table 3.5: Bounding-Region Metric Values in the Two Dimensional Case, for the Z-Order, Hilbert, H-Order, AR^2W^2 , $\beta\Omega$, Peano, and Balanced Peano Curves. Results are Taken From the Work by Haverkort and van Walderveen, where more extensive results are given [2].

	WBA	*WBP*	*ABA*	*ABP*
$\mathcal{Z}$	∞	∞	2.92	2.40
$\mathcal{H}$	2.400	2.400	1.44	1.19
H	3.000	3.000	1.78	1.42
$\mathcal{A}$	3.055	3.125	1.49	1.22
$\beta\Omega$	2.222	**2.250**	1.42	**1.17**
$\mathcal{P}$	**2.000**	2.722	1.44	1.28
Balanced Peano	**2.000**	2.155	**1.44**	1.19

For the three-dimensional case, Haverkort shows how there are various Hilbert-like SFCs that have better properties than the canonical Hilbert Curve [62], [94]. Furthermore, the Hilbert-like SFCs have differing Bounding-Region metric values. For all Hilbert-like curves, they show computationally that the Worst-case Bounding-Box Volume WBV and Surface Area WBS fall within the following ranges:

$$3.11 \lessapprox WBV \lessapprox 14.11 \tag{3.47}$$

and

$$2.54^{\frac{3}{2}} \approx 4.048 \lessapprox WBS \lessapprox 15.96 \approx 6.34^{\frac{3}{2}}. \tag{3.48}$$

They further refine these bounds for a subset of Hilbert-like curves, called Symmetric Face-Continuous SFCs, which are symmetric along a given dimension and have each adjacent sub-hypercube share a two-dimensional face. The equivalent bounds for these Hilbert-like curves are

$$3.11 \lessapprox WBV \lessapprox 3.73 \tag{3.49}$$

and

$$2.99^{\frac{3}{2}} \approx 5.170 \lessapprox WBS \lessapprox 6.436 \approx 3.46^{\frac{3}{2}}. \tag{3.50}$$

One should notice that the bounds for WBS are to the power of $3/2$. Though not explicitly stated by Haverkort [94], these exponents are used to counteract the effect of r in the three-dimensional Bounding-Box Surface Area definition (3.31). In the definition provided in Section 3.2.1, r is used to normalize the scaling as a function of the sidelength N: $r = \frac{d}{d-1}$. However, the scaling of WBS as a function of the number of dimensions results in higher values for three dimensions than for two dimensions. Therefore, Haverkort calculates $WBS^{2/3}$, $WS^{2/3}$, and WBV. The Bounding-Box Volume scales similarly to the denominator $|x_i - x_j|$ in (3.32) and thus no additional exponent is necessary. Even though they utilize this extra scaling method, given that the definitions provided by Haverkort in [62], [94] do not utilize these exponents but only in their computational solutions, they are not utilized here.

Table 3.6: Bounding-Region Metric Values in the Three Dimensional Case, for the Alfa and Beta Curves. Results are Taken From the Work by Haverkort, where more extensive results are given [94].

	Alfa	Beta
WBV	3.11	3.14
WBS	4.41	4.05
$WBS^{\frac{2}{3}}$	2.69	2.54

Haverkort concludes that there are two Hilbert-like SFCs with good locality preserving properties: the Alfa and Beta curves [94]. Their metric values are given in Table 3.6. Comparing their metric values to the bounds shown in Eqs. (3.47) to (3.50), it is seen that the Alfa Curve achieves the best Worst-case Bounding-Box Volume *WBV* and the Beta Curve achieves the best Worst-case Bounding-Box Surface Area *WBS* [94].

Even though Haverkort investigate SFCs in three dimensions [62], [94], they do not apply the same metrics to the non-Hilbert Curves covered in the two-dimensional case [2]. In their work on the Bounding-Region metric for two dimensions, Haverkort and van Walderveen find that the Balanced Peano Curve has the best *WBA* and *WBP* values of 2.000 and 2.155 respectively [2]. This is better than the canonical two-dimensional Hilbert Curve which has $WBA = 2.400$ and $WBP = 2.400$. This leaves open the possibility that there may be a three-dimensional variant of the Balanced Peano Curve that performs better than any of the Hilbert-like SFCs covered by Haverkort. Of the SFCs covered by the other metrics in Sections 3.1 and 3.2, the $\beta\Omega$, H-Order, and Peano Curves do reasonably well. However, the Z-Order Curve has infinite Worst-Case and much higher Average-case metric values; indicating it is not a well performing SFC for this type of metric.

3.3.5 Clustering Metric

The Clustering metric entails two aggregations: the maximum and average of the number of clusters for all region queries on an SFC. Bounds on these metrics are shown in Fig. 3.5, with respect to the SFC sidelength N and the region size parameter r_s. Asano *et al.* show that RSFCs can resolve square region queries with at most 4 clusters, if a constant multiple of extra d-dimensional values can be accessed [31]. In this case, for a square region of $r_s \times r_s$ elements, Cr_s^2 actual points would be retrieved. They make this modification as they utilize the Clustering metric to quantify how many disk-seeks are required to resolve a range query on SFC-mapped data stored on a magnetic medium, such as a hard-disk. They define the $\mathrm{AR}^2\mathrm{W}^2$ Curve, which they prove has the optimal number of clusters for any RSFC of $H_{\max}\big(\mathcal{A}_m^2, \square\big) = 3$ [31].

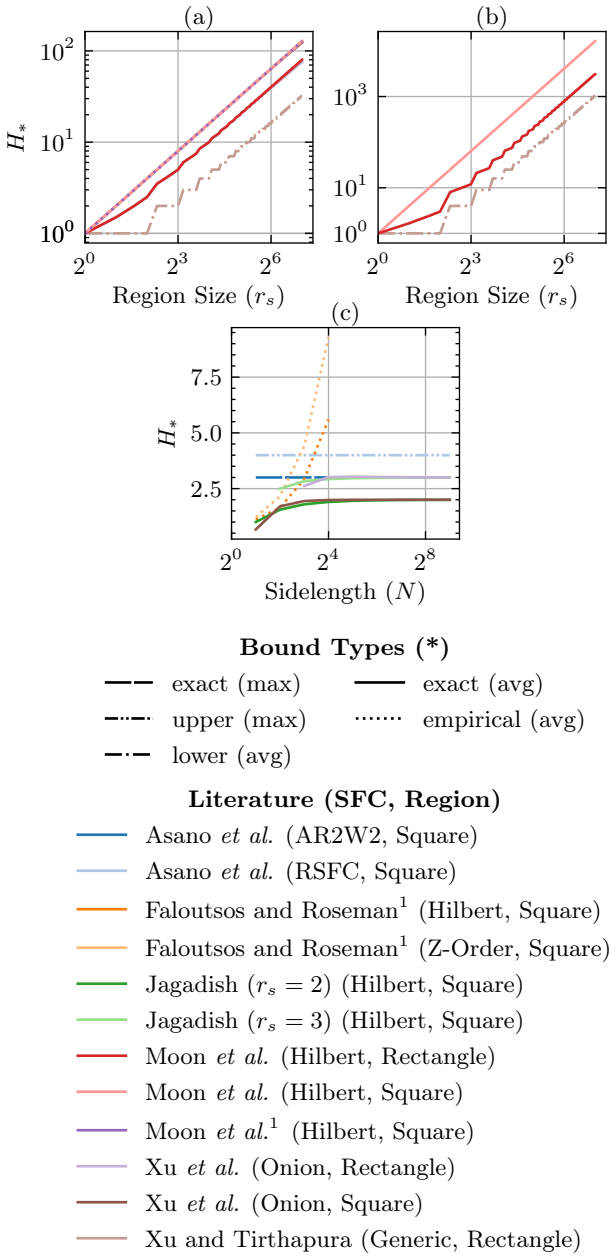


Figure 3.5: Some bounds for the Clustering metric. The subfigures denote different parameters and variables against which the metric is plotted: (a),(b) plotted against the region size parameter r_s, (c) plotted against the sidelength N, (a), (c) $d = 2$, and (b) $d = 3$. Plots for rectangular regions have $r_s/4$ sidelengths except for one which is equal to r_s. Xu *et al.* plots in subfigure (c) have $r_s = 2$ for square regions. Rectangular regions in subfigure (c) have $r_s = 4$ and $r_s/2$ for all region sidelengths except for one which is r_s. Faloutsos and Roseman plots in subfigures (a) and (b) are for all possible region queries in the given SFC space. [1]Bounds are found empirically.

It is evident in Fig. 3.5 that the average number of clusters for a fixed-sized region query converges to a finite non-unity number as the sidelength of the SFC increases. The *divergent* empirical results by Faloutsos and Roseman differ as they are computed for all possible region sizes r_s [11]. The exact bounds for the average Clustering metric H_{avg} in the two-dimensional case for fixed-sized regions shows two things: the performance of the Hilbert and Onions Curves are similar for $r_s << N$. Jagadish shows this for square regions, $r_s \in \{2, 3\}$, and the Hilbert Curve [59]. Xu, Nguyen, and Tirthapura define the Onion curve which they claim has better clustering than the Hilbert curve, which is not immediately evident in Fig. 3.5 [33]. Firstly, one must observe that the Onion Curve plot for square regions in Fig. 3.5(c) differs for small sidelengths. The principle property of the Onion Curve that leads Xu, Nguyen, and Tirthapura to make this claim is that the average number of clusters for the Onion Curve decreases as the region-size is increased beyond $r_s > \frac{N}{2}$. This can be seen in Fig. 3.5(a), where the Xu, Nguyen, and Tirthapura (Onion, Rectangle) and (Onion, Square) plots are the same as those from Moon *et al.* (Hilbert, Rectangle) and (Hilbert, Square) but deviate slightly around $r_s = 2^6$. The number of clusters for a given region on the Hilbert Curve increases with increasing sidelength [33], [35], [108] whereas the number of clusters for the Onion Curve peaks at around $r_s = \frac{N}{2}$. Given that the plots use the logarithmic scale, this is not as visible.

A major contribution by Xu and Tirthapura and Moon *et al.* is that the number of clusters for a given region query is proportional to the surface area of the region itself [35], [40]. In fact, the analytical results plotted in Fig. 3.5 are at times the surface area of the region exactly. Another contribution by Xu and Tirthapura is in identifying the lower bound of the average Clustering metric, for any SFC. As can be seen in Figs. 3.5(a) and 3.5(b), there is still a significant difference between the exact average Clustering metric for the Hilbert [35] and Onion [33] Curves and the lower bound for all SFCs.

Furthermore, Xu and Tirthapura identify the relationship between the number of clusters and rotating the query regions [40]. Rotations are defined as the permutation of the region sidelengths or any axis-aligned rotation. The lower bound they identify, for any SFC for rectangular regions, assumes that no rotations are used [40], whereas the work by Moon *et al.* on the Hilbert Curve and rectangular regions implicitly

assumes all rotations are used [35]. If only continuous SFCs are taken, this lower bound is the exact bound, a guarantee that the curve is optimal for all rotations of rectangular regions [40].

The Onion Curve is continuous, and therefore the advantage it has over the Hilbert Curve is restricted to regions with no rotations. The step-changes or oscillations in Figs. 3.5(a) and 3.5(b) are caused by the plotting method used. Given that a rectangular region cannot have real sidelengths, the ceiling had to be taken. For a given region size parameter, the rectangular regions used to plot have sidelengths $r_s \times \lceil \frac{r_s}{4} \rceil \times \frac{r_s}{4}$. For the sidelength plot, the denominator is set to 2 and r_s is set to 2 and 4 for square and rectangular regions respectively.

Whether one SFC achieves *better* clustering, using the Clustering metric, is dependent on three components: whether more points may be retrieved for the query, if the region size parameter is fixed or not, and whether all rotations of the region are included. The AR^2W^2 Curve appears to be the best if you only have the first of these components [31]. If the application utilizes a fixed-size region, then the Hilbert Curve may be the best for $r_s << N$ [35] and the Onion Curve for $r_s > \frac{N}{2}$ [40]. If the region is fixed in size, but all rotations are taken, then any continuous SFC is optimal.

3.4 Applications of Locality Metrics

Some important applications of these metrics and SFCs are discussed here, with a focus on how the metric *measures* a property relevant to the computational problem. Though there are far more applications than those covered here, these are selected from the metric literature directly.

3.4.1 Warehouse Logistics

There are two applications in logistics where locality preservation metrics are applicable, both being how to place items in a warehouse to be retrieved in the shortest

timeframe. Hua and Zhou investigate using SFCs to order a circuit board assembly kitting area, which stores components for electronic devices [111]. The application involves a picker traversing throughout the warehouse to retrieve certain components on a bill-of-materials. Only a certain number of components can be carried at a time and some components come in reels which must be returned to their original location within the warehouse after the correct quantity is dispensed. They identify a *clustering* type metric to quantify the SFC performance. However, it is useful to understand this problem with respect to the other logistics application.

Some warehouses store items far larger than those used in circuit boards. In some of these instances, overhead cranes are used to transport these items. In the first application, a *picker* must travel along the SFC to retrieve the components. The crane, however, may have a motor for each axis and thus the time it takes to travel between two points is the maximal distance $\|\cdot\|_\infty$. The distance a *picker* has to travel is equivalent to the length of the curve between their start and end points, which is equivalent to the partition along the SFC. However, the optimal distance for a *picker* to retrieve an item is the Euclidean Distance: i.e. when there is no prescribed path within the warehouse.

Therefore, the Hölder Continuity Ratio is the best metric for evaluating the performance of an SFC for the retrieval of items in a warehouse. Where a *picker* must travel along the curve to acquire each item, the $p = 2$ variant should be used: $G_2^{\max}$. Where an overhead crane is used, the $p = \infty$ variant should be used: $G_\infty^{\max}$. This metric only evaluates the worst-case impact of the SFC traversal, and not the average. Modifying the aggregation to include the average of the Hölder Continuity Ratio would give some insight into the overall performance of the SFC for the application.

3.4.2 Scientific Computing

In many multidimensional computational problems, the data are stored in a file mapped using Row-Major Order. An SFC may be applied, in which case the values of the data are still stored in a single dimension in the file, but now in an order defined by the curve. For certain multidimensional algorithms, a local context is

needed. An example of this is a differential operator. The distance in the file it must traverse before retrieving the values is equivalent to the EdgeSum for the operator's context [100]. However, if it is known that the operator requires a given region of data from the file, then multiple values can be retrieved. In this scenario the Clustering metric is more appropriate than the EdgeSum, depending on the method with which the operator is applied.

The application of SFCs to numerics goes further, with research being conducted into how they can improve the performance of Finite Element Modelling analysis (FEM) [95], [104] and Computational Geometry [96]. The Logarithmic EdgeSum and the Partition Ratio metrics are appropriate for this application. The former can be used to quantify the search time-complexity for a candidate SFC, as many search algorithms scale with an order of $O(\log n)$. For FEM, it is not the search time that is the issue, but more the ability to partition the model into regions. The Partition Surface Ratio metric is the most appropriate in this scenario as the surface area of a partition in the FEM space is a key factor in the performance of the associated FEM algorithms [95], [96]. Dividing a FEM system into partitions, to compute in parallel, requires communication between each thread or device assigned to each partition. Therefore, to reduce the overhead of this communication step, the boundary area of all partitions should be reduced. The Partition Surface Ratio metric quantifies this property exactly [95], [96].

These aforementioned applications deal with the ability to divide the computational problem into separate sets which can be executed in parallel. Chochia, Cole, and Heywood state that the performance of a Parallel Random-Access Machine, utilizing SFCs, can be predicted using the Hölder Continuity Ratio [88]. For a given computing system, the communication pathways are very important in determining the performance of parallel algorithms. Korneev and Shchepin show how the partitioning of a multidimensional lattice, for a parallel computing system, is best done using the Maximal Norm Hölder Continuity Ratio $G_{\infty}^{\max}$ [92]. The choice of $p = \infty$ is dependent on how the lattice is partitioned, how partitions can be accessed, and the networking infrastructure used by the computing system.

Though the HCR metric appears best for these applications, the best metric for predicting performance is highly dependent on the algorithms used. For example, Deford and Kalyanaraman define the Average Communicated Distance (ACD) metric from the Nearest Neighbour Stretch (NNS) to quantify the impact of SFCs for N-Body problems [110]. The NNS is used as the basis for the ACD metric as the authors choose to measure the distance a parallel processor must communicate, rather than the ratio of the same distance and the SFC index-difference (which would be the HCR metric).

Another application of the EdgeSum metric is to the Minimum Linear Arrangement (MinLA) problem, or Wire Length problem, which involves ordering the nodes of a graph. Very Large-Scale Integration design involves solving this problem to optimize the layout of subcomponents [112].

3.4.3 Data Storage and I/O

As is discussed in Sections 3.2.2 and 3.3.5, the Clustering metric is meant to measure the number of disk seeks to retrieve a given region query [11], [31], [35], [59]. If data are mapped to disk using an SFC, then a region query in the d-dimensional space would require multiple contiguous regions of the storage medium to be accessed. These clusters require the physical storage medium to seek to subsequent clusters. For magnetic hard disks, this is done by moving the read-write head to different sectors. This is process is also applicable for other non-random-access storage technologies.

3.4.4 Multidimensional Data and Algorithms

Processing multidimensional data typically requires special spatial data structures, such as R-Trees [113] or Quad-Trees [114]. The former stores point data in buckets constrained by bounding boxes, which may overlap. The latter stores point data in similar buckets, which do not overlap and are defined as subquadrants of the full d-dimensional space. One spatial computational problem that utilizes these data structures is the Nearest Neighbour Search. Given that R-Trees have bounding boxes

of different sizes that may overlap, the Bounding-Region metric [2], [62], [94] is the most appropriate for quantifying the locality preservation of an SFC-mapping applied to the data structure [4], [64]. The Bounding-Region metric is not appropriate for the Quad-Tree, as the bucket-regions do not differ in size or overlap [114].

The Irregularity Vector is a good metric for applications where some degree of sorting is required [106], [107]. Examples of this are QoS Routing and some multimedia databases. This metric has the added benefit of identifying SFCs that best suit any intentional bias to a given dimension in the application. Where sorting is not needed, the Description Vector metric is a suitable replacement for the Irregularity Vector as it can also be used to match the intentional bias and fairness of SFCs and the application [36], [105].

One scenario where these metrics are useful is in the compression of multidimensional grid data. Some compression schemes encode the data as the difference between adjacent values [52]. Others compress by removing repeated values or sequences [41], [42]. In both cases, compression is improved if similar values are in close proximity to each other along a given traversal of the data. However, any given multidimensional dataset has its own statistical properties that define how similar values are in each dimension: a dimensional bias in the correlation of values. Some datasets are isotropic, and thus have no dimensional bias [72]. Using the Description and Irregularity Vectors' notions of fairness and intentional bias, the degree of isotropic behaviour for a dataset and the intentional bias of an SFC can be matched. This should increase the correlation of values, and thus increase compressibility. A similar technique is used by Ouni, Lassoued, and Abid where they develop a Gradient-based SFC to increase the data correlation when mapped, which they show improves compression performance [47].

3.5 Conclusion from Analysis of Metrics

This paper reviews locality preservation metrics utilizing a measurement-aggregate-metric methodology. Six broad metric types are covered, categorized, and compared; with relationships identified between their definitions and usage in the literature.

Metrics are grouped into two overarching types, Point-Pair and Region metrics, with four and two subtypes each. Alongside this categorization of locality preservation metrics, an analysis of analytical and empirically bounds — from the literature — on metric values for specified SFCs is given. Through this analysis, scenarios where certain curves are *best* are identified: either as confirmation of conclusions from the literature or as a comparison of said conclusions. A further contribution of this work is in the analysis of applications where the locality preservation metrics can be appropriately used, alongside SFC mappings. Through this, it is shown how the measurements for each metric are related to specific properties of the applications. Appropriate applications for the use of these metrics include warehouse layout optimization, scientific and parallel computing, data storage optimization and performance evaluation, spatial data manipulation and retrieval, search and sort operations, and multidimensional data compression.

Chapter 4

Effect of d-Dimensional Re-orderings on Lossless Compression of Radio-Astronomy and Digital Elevation Data

A paper from this research, covering the empirical experiments and quantitative analysis of SFC-based compression methods, was published in the IEEE Access journal on the 28th of May 2021 and is given here. Within this chapter the DEM and radio-astronomy datasets are analysed, the compression pipeline from which results are obtained is discussed, and an analysis of the performance of SFC-based compression schemes are given. This paper and chapter form the core contribution of this work; addressing the second and third research questions in Section 1.1 and second through fifth objectives in Section 1.3. The citation for this paper is

> **C. J. Haupt**, E. J. Otoo, and L. Cheng, "Effect of d-Dimensional Re-orderings on Lossless Compression of Radio-Astronomy and Digital Elevation Data," *IEEE Access*, vol. 9, pp. 80 415–80 433, 2021, ISSN: 2169-3536. DOI: 10.1109/ACCESS.2021.3084838. [Online]. Available: https://ieeexplore.ieee.org/document/9443202.

Sections relevant to this paper, whose content is included in the paper manuscript, are Sections 2.1, 2.4 and 2.5. The core of the research in this paper was conducted by **C.J. Haupt**, under the guidance and supervision of Prof. E.J. Otoo and Prof. L. Cheng. The principle research idea was identified in a joint effort between **C.J. Haupt** and Prof. E.J. Otoo. The data preparation, software implementation, experiments, and results analysis was implemented/conducted by **C.J. Haupt**. The paper was written by **C.J. Haupt** under the supervision of Prof. L. Cheng, ensuring the quality and standard necessary for international publication was achieved.

4.1 Methodology

To apply the SFCs, BitShuffle, and compression schemes to the two datasets, the *C++* program `sfccompress` was developed with an accompanying Python script, `sfcc_compress.py`, for automation. Both facilitate the reordering of SFCC input data using either the Raster Scan, Z-Order, Gray-Code, or Hilbert Curves, the application of BitShuffle, and the compression of the resulting preprocessed data. BZIP2, GZIP, LZ4, and LZO — being *external* tools — are applied separately to the `sfccompress` program. The `sfcc_compress.py` script acts as an umbrella, executing the appropriate tools to apply a given configuration to a dataset file. The full compression process is illustrated in Fig. 4.1. The `sfccompress` program operates on SFCC files which contain raw data from the two datasets and metadata to facilitate compression, as discussed in Section 2.4.

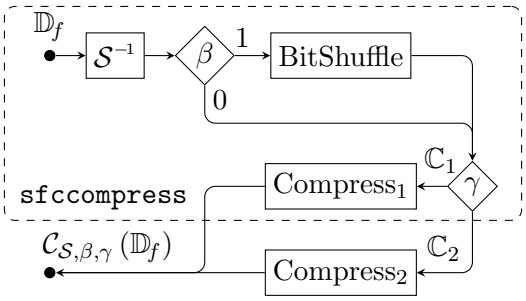


Figure 4.1: Compression process utilizing `sfccompress` and external compression tools. Internal and external compression methods/tools are labelled as $\mathbb{C}_1$ and $\mathbb{C}_2$ respectively. $\beta = 1$ when BitShuffle is applied and 0 otherwise. γ indicates which compression scheme is to be used. The compressed output for an input file $\mathbb{D}_f$ is $\mathcal{C}_{\mathcal{S},\beta,\gamma}(\mathbb{D}_f)$. Decompression is carried out in the reverse order.

The process is parametrized using $\mathcal{S}$, β, and γ, representing the chosen SFC reordering, whether BitShuffle is applied (1 for yes, 0 for no), and the compression scheme to use, respectively. The compressed version of an SFCC file $\mathbb{D}_f$, for a given configuration, is given as $\mathcal{C}_{\mathcal{S},\beta,\gamma}(\mathbb{D}_f)$. As an example, compressing a file with the Hilbert Curve, BitShuffle, and GZIP gives $\mathcal{C}_{\mathcal{H},1,\text{GZIP}}(\mathbb{D}_f)$. Execution of the compression and decompression processes is conducted using GNU Parallel [115] with four

threads, the number available on the computing hardware used. An input list of parameters and filenames is supplied to GNU Parallel which passes them on to `sfcc_compress.py` where the appropriate tools are called. Timing is measured in milliseconds by GNU Parallel. The ordering of parameters and filenames is such that decompression jobs cannot occur before the compressed output is created in a previous job. As the process is lossless, the validity of each step was verified by reversing the process and confirming the final output matches the initial input using a SHA256 hash-sum. The full compression and decompression process was conducted over two days on a desktop computer with an Intel Core i5-2500K CPU, 16 GB of 1333 MHz DDR3 memory, a 500 GB Seagate Barracuda SSD with EXT4 partitions, the GNU C Compiler (GCC) version 9.3.0, and Ubuntu 20.04 with version 5.4.0-47 of the Linux kernel.

Storage requirements for each file is measured in bytes using the `du` Linux tool. Compression and decompression speed is calculated as megabytes-per-second (MB/s) by dividing the size of each dataset, uncompressed, by the sum of execution times for a given configuration. The uncompressed size is used for both the compression and decompression speeds. The runtime is measured as the number of milliseconds elapsed from the start of the *sfcc_compress.py* script execution to the return of an exit code. Compression ratios are calculated as the ratio of the uncompressed size and the compressed size (see (2.18)). Other measurements made include statistical properties of the preprocessed files using Python. File entropy is measured on a byte level, normalized between zero and one. The block-entropy of each preprocessed file is calculated with a blocksize of 16 KiB, also normalized between zero and one.

4.1.1 Implementation Details

The Raster Scan mapping in `sfccompress` is implemented using modulo arithmetic on integer variables. For the Z-Order and Gray-Code Curves, *libmorton* is used [116]. Unfortunately, the CPU used does not support the BMI2 or AVX2 instruction sets, which *libmorton* can use to improve mapping speeds. However, this did not appear to be an issue with the given dataset sizes. The Gray-Code Curve mapping implementation uses the same *libmorton* calls but with six bitwise operations to translate

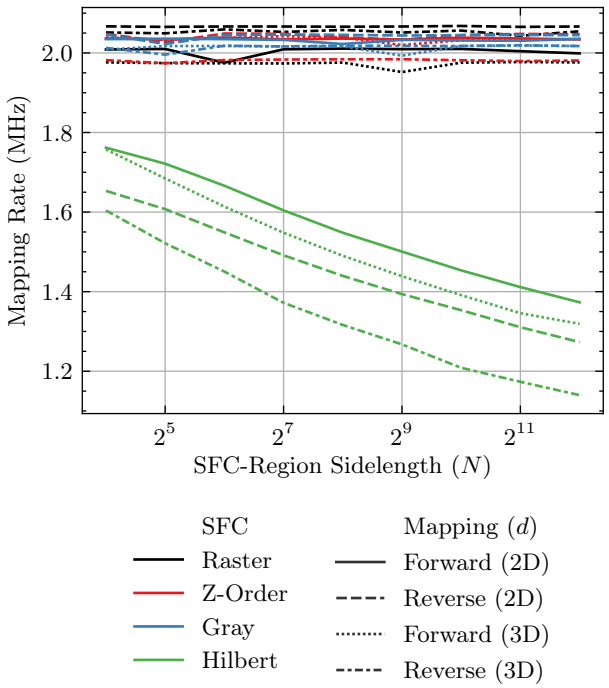


Figure 4.2: Mapping Rate for the four SFCs on the benchmarking computer. The sidelengths for the 2D SRTM dataset is 2048 (2^{11}) and the 3D SKA dataset is 128 (2^7).

between the Gray-Code index and value. The mapping algorithm by Chenyang, Hong, and Nengchao [117] is used for the Hilbert Curve components of `sfccompress` by integrating the *libhilbert* library [118]. The SFC mapping rates from `sfccompress` are shown in Fig. 4.2. The Hilbert mapping is the slowest by a significant margin while the other three SFCs achieve similar mapping rates. Furthermore, the Raster, Z-Order, and Gray-Code mappings are not affected by the increase in sidelength. This is because the algorithm by Chenyang, Hong, and Nengchao iterates through every recursive subregion of the Hilbert mapping and is thus dependent on the SFC order m and the sidelength $N = 2^m$ [117]. However, the other three SFC mappings are not dependent on the sidelength while $N^d \leq 2^{64}$ as their mapping algorithms operate on 64-bit integer variables in the same manner irrespective of the actual value of N. For significantly larger sidelength values, more integer variables would be needed by the algorithms, incurring a small runtime penalty.

The implementation of BitShuffle only conducts a bit-level reorganization of the underlying values, as described in Section 2.5.1, and not the lossy precision reduction defined by Masui *et al.* [20]. This BitShuffle implementation is coded in *C++* and integrated into the `sfccompress` program. It is important to note that the BitShuffle process is dependent on the data type being used but not the number of dimensions. For the SRTM and SKA datasets, the values are 16 bit signed integers and single-precision floating-point values respectively, corresponding to two and four bytes each. For an n bit data type, blocks of n values are shuffled at a time, packing their bits into the output array based on their bit-level significance. Therefore, a single block processes $\frac{2\times n^2}{8}$ bytes at a time: 64 B for *int32* and 256 B for *float*. The performance of this process is shown in Fig. 4.3. The shuffle rate quantifies how many n bit input values are *shuffled* per second while the size of the SFC space is the size of a given data file, excluding all metadata ($\frac{n}{8} \times N^d$).

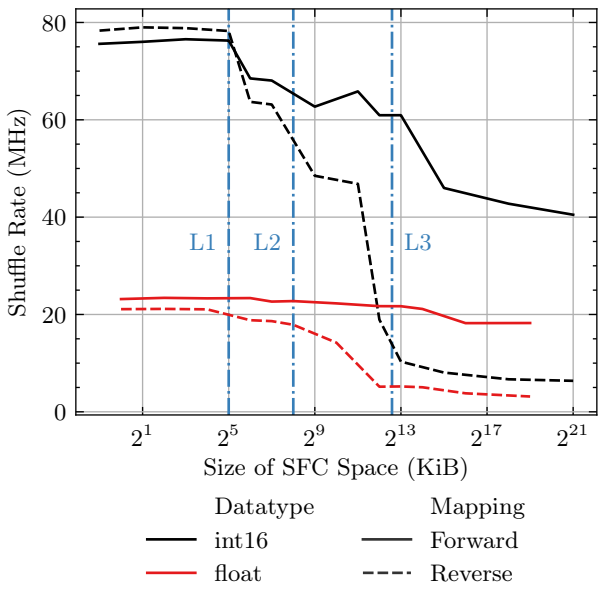


Figure 4.3: BitShuffle implementation shuffle rate vs the size of the data being mapped. All files compressed in this paper have a 2^{13}KiB sized SFC space. Note that the SRTM dataset values are 16 bit integers and the SKA dataset values are single-precision floating-point values. The L1, L2, and L3 cache sizes are shown in blue: 32 KiB, 256 KiB, and 6144 KiB respectively. The shuffle rate is measured in values per second.

The sizes of the L1, L2, and L2 caches for the Intel i5 CPU used are shown in Fig. 4.3 in blue. The shuffle rate for both the forward and reverse mappings decreases when the SFC Space exceeds the L1 Cache size but settles at a floor after the L3 Cache size. All four cores in the i5 2500K CPU share the L3 Cache whereas there are dedicated L1 and L2 Caches for each of the four cores. However, the BitShuffle implementation is not parallel and therefore only one L1 Cache and one CPU core are used. The float data type has a shuffle rate of about a quarter of the 16 bit integer data type. Normalizing for data type size, BitShuffle on single-precision float data is still half as fast as with the smaller data type. For a given size of the SFC space, different data types result in different access patterns for the input and output arrays. Therefore, it is hypothesized that the slower shuffle rate for the float data type is primarily caused by cache misses in the BitShuffle *C++* implementation used. Secondly, the greater decrease for the reverse mapping compared to the forward mapping may also be attributed to cache misses and out-of-cache memory accesses. The memory *read* access pattern for the forward mapping is the memory *write* access pattern for the reverse mapping, and vice versa.

4.2 Experimental Results

Results are separated into three categories — compression ratios, compression and decompression speeds, and information entropy — where both datasets are covered. There are two important baseline configurations for the compression ratio and speed results. The most common in real-world applications is that without SFC reordering and without BitShuffle — equivalent to compressing the data without any preprocessing: this is the *default* scenario. We refer to this as the unpreprocessed data as no preprocessing has been carried out on the dataset values. The second baseline is when the Raster Scan is used, as is the case with the former, but with BitShuffle applied. This scenario is mostly applicable to the SKA dataset as BitShuffle is compatible with HDF5, the format in which the data were acquired, and was designed for the type of data present in the SKA dataset. Therefore, it is likely that BitShuffle is already incorporated into other radio-astronomy datasets. However, none of the DEM file-formats support BitShuffle, and it is less likely that BitShuffle would be

applied to such datasets. In each of the following subsections the key observations are identified and important comparisons raised. A discussion of the results is given in Section 4.3. Empirical compression ratio and speed results for all configurations are given in Tables 4.1 and 4.2.

Table 4.1: Empirical Results for all Combinations of Compression Schemes, BitShuffle, and SFC Reorderings with the SRTM Dataset. Measurements Shown are Compression Ratio $\mathcal{R}_c$, Compression Speed, and Decompression Speed. The Highest Result for a Given Combination of BitShuffle β, Compression Scheme, and Dataset is in Bold and Underlined. Where all Values are the Same, None are in Bold or Underlined. Note that β Denotes Whether BitShuffle [20] is Applied ($\beta = 1$) or not ($\beta = 0$). All Results are Shown to Three Significant Figures. Compression Schemes Shown are BZIP2 [78], GZIP (DEFLATE) [51], [119], Huffman Encoding (HFF) [45], LZ4 [56], LZ77 [41], LZO [81], LZW [44] (a Variant of LZ78 [42]), and Run-Length Encoding (RLE) [43].

			SRTM Dataset							
			BZIP2	GZIP	HFF	LZ4	LZ77	LZO	LZW	RLE
Comp. Ratio ($\mathcal{R}_c$)	$\beta = 0$	**Raster**	**6.54**	3.52	1.44	**3.18**	**2.70**	2.96	2.39	**1.13**
		Z-Order	5.25	3.52	1.44	2.94	2.52	3.04	3.09	1.05
		Gray	5.04	3.45	1.44	2.88	2.46	3.00	3.03	1.05
		Hilbert	5.94	**3.76**	1.44	3.13	2.68	**3.21**	**3.28**	1.11
	$\beta = 1$	**Raster**	4.02	4.06	1.98	3.67	3.51	3.70	3.21	2.13
		Z-Order	4.40	4.31	**2.05**	3.97	3.99	3.97	3.42	2.41
		Gray	4.38	4.29	**2.05**	3.94	3.95	3.96	3.41	2.40
		Hilbert	**4.47**	**4.33**	2.04	**4.02**	**4.06**	**4.03**	**3.45**	**2.46**
Comp. Speed (MB/s)	$\beta = 0$	**Raster**	7.43	**6.05**	**24.7**	**6.58**	**15.6**	**3.00**	**7.07**	**40.0**
		Z-Order	7.34	3.16	17.7	5.65	11.9	1.74	6.86	29.1
		Gray	**7.48**	3.24	17.3	5.78	11.6	1.79	6.83	27.0
		Hilbert	3.56	2.09	4.99	3.02	4.63	1.36	3.45	5.49
	$\beta = 1$	**Raster**	**8.88**	4.81	**22.9**	10.8	**9.39**	2.79	**7.20**	**30.8**
		Z-Order	7.93	**5.36**	16.7	**11.0**	8.83	**3.22**	6.84	22.2
		Gray	7.63	5.11	16.2	10.4	8.90	3.11	6.89	23.0
		Hilbert	3.71	2.89	4.81	4.27	4.14	2.19	3.42	5.23
Decom. Speed (MB/s)	$\beta = 0$	**Raster**	**7.43**	**6.01**	**23.4**	**6.58**	**15.5**	**2.98**	**7.10**	**30.4**
		Z-Order	6.91	3.07	17.6	5.29	11.3	1.71	6.88	21.5
		Gray	6.51	3.04	17.0	5.17	11.2	1.73	6.70	21.1
		Hilbert	3.51	2.08	4.89	2.98	4.62	1.36	3.40	5.29
	$\beta = 1$	**Raster**	**7.58**	4.40	**20.2**	**8.94**	**9.20**	2.64	**7.19**	**26.8**
		Z-Order	6.71	**4.81**	14.6	8.92	8.66	**3.00**	6.73	21.7
		Gray	6.69	4.61	15.8	8.67	8.71	2.91	6.89	21.3
		Hilbert	3.46	2.74	4.79	3.92	4.13	2.11	3.37	5.19

Table 4.2: Empirical Results for all Combinations of Compression Schemes, BitShuffle, and SFC Reorderings with the SKA Dataset. This caption is equivalent to that for Table 4.1.

			SKA Dataset							
			BZIP2	GZIP	HFF	LZ4	LZ77	LZO	LZW	RLE
Comp. Ratio ($\mathcal{R}_c$)	$\beta = 0$	**Raster**	1.06	1.10	0.941	**1.03**	**1.03**	**1.03**	0.822	0.800
		Z-Order	**1.07**	**1.12**	0.941	**1.03**	**1.03**	**1.03**	0.834	0.800
		Gray	**1.07**	**1.12**	0.941	**1.03**	**1.03**	**1.03**	0.835	0.800
		Hilbert	1.06	1.10	0.941	1.01	1.01	1.01	**0.840**	0.800
	$\beta = 1$	**Raster**	1.20	1.19	0.915	1.16	1.17	1.16	0.884	0.820
		Z-Order	**1.23**	**1.25**	**0.952**	**1.22**	**1.24**	**1.22**	**0.905**	**0.882**
		Gray	**1.23**	1.24	**0.952**	**1.22**	**1.24**	**1.22**	**0.905**	0.868
		Hilbert	1.21	1.20	0.948	1.19	1.21	1.19	0.891	0.876
Comp. Speed (MB/s)	$\beta = 0$	**Raster**	4.14	10.4	**18.7**	12.1	**2.44**	6.31	**3.21**	**35.9**
		Z-Order	4.25	11.4	15.9	14.3	2.41	**6.86**	3.16	32.5
		Gray	**4.36**	**11.5**	14.8	**14.4**	2.41	6.75	3.10	33.3
		Hilbert	3.39	6.50	7.79	7.48	2.17	4.67	2.62	10.9
	$\beta = 1$	**Raster**	**4.16**	**6.68**	**14.4**	**10.8**	2.81	**3.68**	**3.23**	**24.3**
		Z-Order	4.07	5.93	12.6	9.84	**2.86**	3.34	3.19	19.2
		Gray	4.14	5.83	12.6	9.66	2.85	3.42	3.20	21.5
		Hilbert	3.36	4.19	6.95	6.16	2.52	2.78	2.64	9.20
Decom. Speed (MB/s)	$\beta = 0$	**Raster**	4.07	10.0	**15.2**	9.80	2.41	6.03	**3.10**	20.9
		Z-Order	4.22	9.94	13.4	11.4	**2.37**	5.98	3.03	**23.2**
		Gray	**4.26**	**10.5**	12.7	**12.0**	2.35	**6.24**	3.05	19.2
		Hilbert	3.37	6.11	7.75	6.56	2.15	4.41	2.59	9.68
	$\beta = 1$	**Raster**	3.93	**5.85**	**11.8**	8.44	2.80	3.40	**3.17**	**17.9**
		Z-Order	4.08	5.81	11.1	**9.01**	**2.88**	3.31	3.14	14.4
		Gray	**4.09**	5.73	10.3	8.81	2.86	**3.41**	3.07	16.4
		Hilbert	3.28	4.04	6.54	5.72	2.51	2.71	2.57	8.75

4.2.1 Entropy Analysis

The SKA dataset has high entropy, with all unpreprocessed and preprocessed files being above 0.9. The SRTM dataset has a broader range of file-entropy values, with the majority lying between 0.25 and 0.65. As the order of values does not play a role in file entropy, the SFCs do not have any effect on their own. When BitShuffle is applied, the byte values used to calculate the file entropy are changed, resulting in different file-entropies. Under this scenario, the Raster Scan results in higher entropy values for the SRTM dataset compared to all three SFCs. The Z-Order, Gray-Code, and Hilbert Curves all achieve similar file-entropy values for the

SRTM dataset whereas the Hilbert Curve deviates from the other two with the SKA dataset. Applying BitShuffle to the SKA dataset, without SFC reordering, increases file-entropies by approximately 0.05. The similarity in file entropy observed between each SFC with the SRTM dataset is not entirely present in the SKA dataset, as the Hilbert Curve with BitShuffle deviates slightly to higher entropy values compared to the Z-Order and Gray-Code Curves. Given that the BitShuffle process reorders by bit and the file entropy is measured on the byte level, the deviation between the three SFCs with the SKA dataset and not with the SRTM dataset can be attributed to two factors.

Firstly, the three dimensions in the SKA dataset have significantly lower autocorrelations compared to the SRTM dataset. Given that the three SFCs traverse subquadrants in different orders, the bit-level statistics of the BitShuffle values differ more with lower autocorrelation values. Secondly, the Z-Order and Gray-Code Curves traverse the least-significant dimension first whereas the Hilbert Curve traverse the most significant first (see Fig. 2.1). Given that the three SFCs are recursive, it is also important to note that each subquadrant of the Hilbert Curve has a different orientation but the other two SFCs do not. Furthermore, Mokbel, Aref, and Kamel show that the Z-Order and Gray-Code Curves have dimensional bias whereas the Hilbert Curve does not [36]. The combination of dimensional-bias in the SKA data, indicated by the autocorrelation and the dimensional bias of the Z-Order and Gray-Code Curves, causes the resulting BitShuffle values to differ from the Hilbert Curve. This is not observed for the SRTM dataset as it is isotropic and thus the dimensional-bias of the curves cannot significantly influence the BitShuffle values and thus the file-entropy.

However, total file entropy is not a good indicator of the underlying file structure. To analyse the impact of SFC reordering and BitShuffle further, the block-entropies for all files are plotted in Fig. 4.4. The similarity between all orderings without BitShuffle for file-entropies is not present in the block-entropies as each block includes different values. The blocksize is of the form 2^{dm} (for $d \in \{2, 3\}$), equivalent to a subquadrant in the SFC mappings. Without BitShuffle, the block-entropies for both datasets are lower with SFC reorderings, though negligible for the SKA dataset ($\approx$0.05 lower). The SFCs reduce the block-entropy for the SRTM dataset without

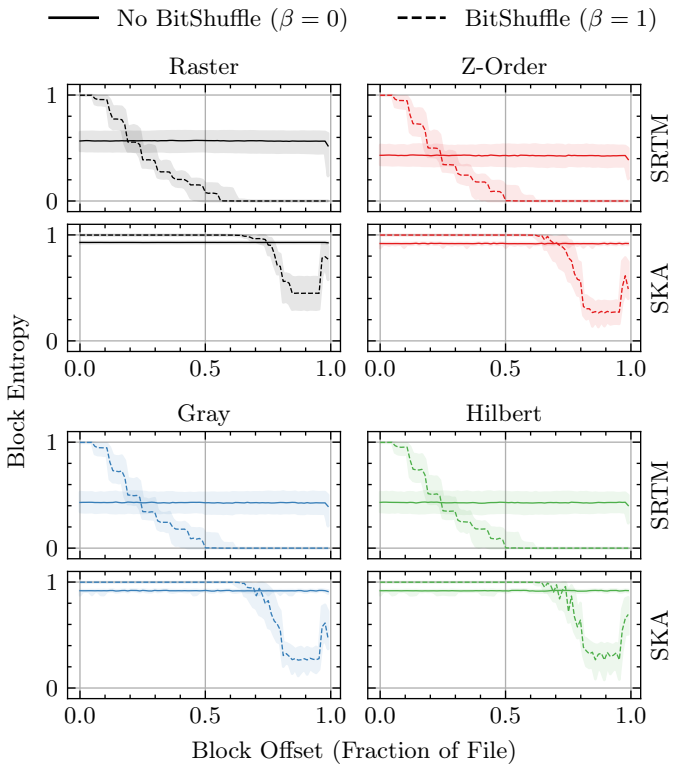


Figure 4.4: Median block-entropy for both datasets, all SFCs, and with and without BitShuffle. The top and bottom rows contain results for the SRTM dataset and SKA dataset respectively. Shaded-areas denote the inter-quartile ranges. The block-size used is 16 KiB. The entropy is calculated using byte-sized symbols and normalized to between 0 and 1.

BitShuffle, compared to the Raster Scan, by approximately 0.15. However, all three SFCs have equivalent block-entropies as they have the same subquadrant extents (all are recursive with $B = 2^d$). Chunking the datasets into blocks with the same extents as the SFC subquadrants (assuming a block-entropy blocksize larger than the chunk size) would result in the same block-entropy results regardless of the SFC used. However, the block-entropies are more useful in understanding the impact of BitShuffle. By showing where the original datasets are the most redundant.

As the block-entropy for the SRTM dataset with BitShuffle is zero for the last 30 % of each file, we can conclude that the most significant 4 bits of the *int16* values are mostly redundant ($\lfloor 30\,\% \times 16\,\text{bit} \rfloor = 4\,\text{bit}$). With SFC reorderings, the median block-entropy is effectively zero for the last half of each file. This implies that at

least half of the SRTM files only require one byte instead of two to represent most of their values when using an SFC and BitShuffle. This would require some metadata to store the actual value of the upper bits for specified blocks of values. With the Raster Scan, the median block-entropy reaches zero approximately one-bit later and thus would require 9 bits when BitShuffle is used.

A similar claim cannot be made for the SKA dataset as the block-entropies never drop to zero. However, the bits in the floating-point values have distinctly different entropies based on their location in the binary representation. Each IEEE-754 single-precision value in the SKA data consists of 32 bits split into a mantissa (21 bits), exponent (8 bits), and sign (1 bits). The first 71.9 % of the BitShuffled SKA files contain the mantissa, with the least-significant bits first. Even though the block-entropy begins to drop at about 60 % through the SKA files, this only accounts for the three most significant bits of the mantissa. The exponent bits are far more redundant, achieving a median block-entropy below 0.5. However, even though SFCs had little effect without BitShuffle, and on the mantissa bits with BitShuffle, the block-entropy for the exponent bits is 0.2 lower with SFC reorderings than with the Raster Scan. As the block-entropy for the mantissa bits is the maximum possible (1.0), it can be assumed that the majority of compression gains on the SKA data are concentrated in the exponent and sign bits.

Not all compression schemes would be able to exploit the lower entropies present in the preprocessed files as they use other non-entropy based algorithms. However, the file-entropies and block-entropies shown are indicative of the restructuring carried out by the SFCs and BitShuffle. Secondly, the significant reduction in block-entropy, for the SRTM dataset without BitShuffle, when SFCs are applied shows that they effectively exploit the high local-correlations of Gaussian Random Fields.

4.2.2 Compression Ratios

For nearly all compression schemes and SFCs, the use of BitShuffle always improves compression ratios. The only two instances where this is not the case is for BZIP2 with the SRTM dataset and Huffman Encoding with the SKA dataset and the Raster

Scan. Compression Ratios are more than 30 % smaller for BZIP2 and the SRTM dataset when BitShuffle is applied. Even though BitShuffle improves compression ratios for the SKA dataset with Huffman Encoding and SFC reordering, the compression scheme always results in larger dataset sizes for the SKA data. LZW and RLE also fail at compressing the SKA dataset to above-unity compression ratios. RLE only achieves $\mathcal{R}_c$ values above two for the SRTM dataset if BitShuffle is used, between 1.05 and 1.13 if not.

Without BitShuffle, RLE has the worst compression ratios for the SRTM dataset whereas it is the worst for SKA dataset with and without BitShuffle. For $\beta = 0$ and all SFCs, RLE has a compression ratio of 0.8 for the SKA dataset owing to the low local correlation of the data and the specific byte encoding used by the RLE implementation. As the SKA dataset uses IEEE-754 single-precision floats, each RLE encoded symbol requires five bytes: a run length and the run value. If there are no runs with lengths greater than one, we would expect a compression ratio of $\mathcal{R}_c = 0.8$, the $\mathcal{R}_c$ achieved for the SKA dataset with RLE and no BitShuffle. This shows that RLE, with this naive encoding scheme, is wholly unsuitable for compressing the SKA dataset. Even with BitShuffle, RLE ratios are only increased by up to approximately 10 % yet still sub-unity ($\mathcal{R}_c < 1$).

Compression Ratios for the SKA dataset never exceed $\mathcal{R}_c = 1.3$. HFF, LZW, and RLE all expand the dataset size — in all cases — rather than reducing it. However, without BitShuffle, the largest compression ratio for the SKA dataset is 1.12 when using GZIP with either the Z-Order or Gray-Code Curves. For the SRTM dataset, all compression schemes achieve $\mathcal{R}_c > 1$, under all configurations. When it comes to the effect of SFC reordering versus the Raster Scan, the curves nearly always improve compression for both datasets when BitShuffle is also applied. Without BitShuffle, the only schemes improved through the use of SFC reorderings are GZIP, LZO, and LZW for the SRTM dataset and BZIP2, GZIP, and LZW for the SKA dataset. Interestingly, the Hilbert Curve gives the biggest improvement of all the SFCs for the SRTM data, when SFCs do improve $\mathcal{R}_c$ values. The Z-Order and Gray-Code Curves do decrease compression ratios by more than 5 %, without BitShuffle, in some cases: BZIP2, LZ4, LZ77, and RLE for the SRTM dataset. The only situation

where the Hilbert Curve reduces $\mathcal{R}_c$ more than 5 % is with BZIP2, no BitShuffle, and the SRTM dataset.

With BitShuffle, compression ratios are increased through the use of SFC reordering anywhere between 3 % and 15.5 % for the SRTM dataset and 0.79 % and 7.56 % for the SKA dataset. These ranges are expanded further when comparing SFC reordering with BitShuffle to the Raster Scan without BitShuffle. The largest increase in compression ratio over the default unpreprocessed row-major ordering is with RLE, BitShuffle, and the Hilbert Curve on the SRTM dataset; increasing $\mathcal{R}_c$ from 1.13 to 2.46, an increase of 117.7 %.

For the SKA dataset, the largest percentage is 20.39 % for both the Z-Order and Gray-Code Curves with BitShuffle and LZ77. Without BitShuffle, the biggest percentage increase using an SFC reordering is 37.24 % and 2.19 % with the Hilbert Curve and LZW for the SRTM and SKA datasets respectively. Interestingly, the Hilbert Curve loses this lead to the other two SFCs when BitShuffle is applied to the SKA dataset. In general, the best SFCs for increasing $\mathcal{R}_c$ are the Hilbert Curve for the SRTM dataset and the Z-Order Curve for the SKA dataset. For the SKA dataset with BitShuffle, the Z-Order Curve improves $\mathcal{R}_c$ by an average of 4.73 % compared to the Raster Scan. For the SRTM dataset, where the Hilbert Curve does improve the compression ratio, $\mathcal{R}_c$ is increased on average by 13.13 % through its use.

The Z-Order and Gray-Code Curves consistently achieve the highest compression ratios for the SKA dataset with BitShuffle, exceeding $\mathcal{R}_c = 1.2$ for BZIP2, GZIP, LZ4, LZ77, and LZO. The Hilbert Curve achieves between 1.19 and 1.21 ratios for the same compression schemes. The largest compression ratio for the SKA dataset with BitShuffle, of 1.25, is achieved with GZIP and the Z-Order Curve. However, the largest absolute increase in $\mathcal{R}_c$ when applying SFC reorderings, for the SKA dataset, is with LZ77 and the Z-Order and Gray-Code Curves, not GZIP; increasing from $\mathcal{R}_c = 1.03$ ($\beta = 0$) and $\mathcal{R}_c = 1.17$ ($\beta = 1$) to 1.24 ($\beta = 1$). The largest absolute increase in $\mathcal{R}_c$ for the SRTM dataset is also for LZ77, but instead with the Hilbert Curve, increasing from 2.96 ($\beta = 0$) and 3.7 ($\beta = 1$) to 4.03 ($\beta = 1$). Without BitShuffle, compression ratios are increased by less than 0.2 for the SKA dataset.

However, the largest equivalent increase for the SRTM dataset is with LZW, from $\mathcal{R}_c = 2.39$ to $\mathcal{R}_c = 3.28$.

For both datasets, the Lempel-Ziv based compression schemes perform similarly. Huffman Encoding and RLE perform the worst while BZIP2 performs the best for the SRTM dataset and on par with the Lempel-Ziv schemes for the SKA dataset. The Burrows-Wheeler Transform (BWT) exploits strong local correlation between adjacent values by implementing a reversible pseudo-sort. If the local correlation is weak, the BWT would be less effective. This is most likely why BZIP2, in which BWT is the *core* component, performs worse for the SKA data.

The aforementioned literature for DEM data achieves higher compression ratios than those shown here. Scarmana and McDougall achieve compression ratios similar to those in the work presented here with LZW and GZIP/DEFLATE but their Hilbert Curve based compression scheme using DPCM achieves compression ratios above $\mathcal{R}_c = 7$, higher than BZIP2 for the SRTM dataset [12]. None of the compression schemes investigated in this paper use predictive coding, the primary technique in DPCM. Furthermore, the dataset used by Scarmana and McDougall is not the same as the SRTM dataset, indicating that the two results are not entirely comparable. The average compression ratio found by Rane and Sapiro for the JPEG-LS compression scheme on DEM data is $\mathcal{R}_c = 11.75$ [55]. This is significantly higher than compression ratios achieved here and in the work by Scarmana and McDougall [12]. Rane and Sapiro investigate only JPEG-LS, the lossless and near-lossless JPEG encoding scheme [55]. Scarmana and McDougall also investigate JPEG-LS with their dataset, but only achieve an average compression ratio of $\mathcal{R}_c = 5.97$ [12], a value much lower than that achieved by Rane and Sapiro for 16 bit integer DEM data [55]. This raises the question of how these approaches to compressing DEM data would perform on a common dataset.

For the SKA dataset, the lossy BitShuffle technique defined by Masui *et al.* attains compression ratios of approximately $\mathcal{R}_c 3.58$, higher than that attained through lossless compression of the SKA dataset [20]. Though, this significant performance is achieved with lossy compression, which is outside the scope of this work. But it does highlight that radio astronomy data is far more redundant when some precision

is lost for the sake of size requirements. The work by Zheng *et al.*, in compressing astronomy data using an *effective bit-width* transform with GZIP, shows that there is still room for lossless compression schemes for similar datasets [58]. Even so, their dataset, and that used by Masui *et al.*, contains integer data and not floating-point data as is the case with the SKA dataset. Regardless of the comparative performance between previous works and the results conveyed here, it is evident that astronomy data requires some bit-level manipulation to compress effectively. Whether this is BitShuffle, an *effective bit-width*, or an alternative scheme, compression is dependent on the underlying bit-level statistics of the data type and dataset.

4.2.3 Compression and Decompression Speeds

The compression and decompression speeds observed are somewhat correlated with the compression ratios achieved. This relationship is evident in Fig. 4.5, where lower $\mathcal{R}_c$ values have higher compression and decompression rates. The numerical results in Tables 4.1 and 4.2 also show how different configurations change processing speeds. Decompression is generally slower than compression by about 20 %, for all compression schemes except RLE, while decompression is up to 26 % and 42 % slower for the SRTM and SKA datasets respectively. However, RLE is the fastest compression scheme in all cases, as no dictionary or encoding table is needed to process the data. Compression speeds are predominantly slower for the SRTM dataset when some preprocessing is applied (SFCs and/or BitShuffle). The application of SFC reordering to the SRTM dataset only improves compression and decompression speeds for GZIP and LZO with BitShuffle. When compared to the unpreprocessed configuration, their improvements are significantly diminished, if not negated entirely. Without BitShuffle, the Raster Scan always performs faster than the other three SFCs for the SRTM dataset.

Nearly all compression schemes, applied to the SKA dataset, are slowed down through the use of BitShuffle. The impact is different for the SRTM dataset where it sometimes benefits the overall speed. In many cases, it has very little impact on compression and decompression speeds for the SRTM dataset. It is possible that the reduced shuffle rate for three-dimensional floating-point data shown in Fig. 4.3 is the cause

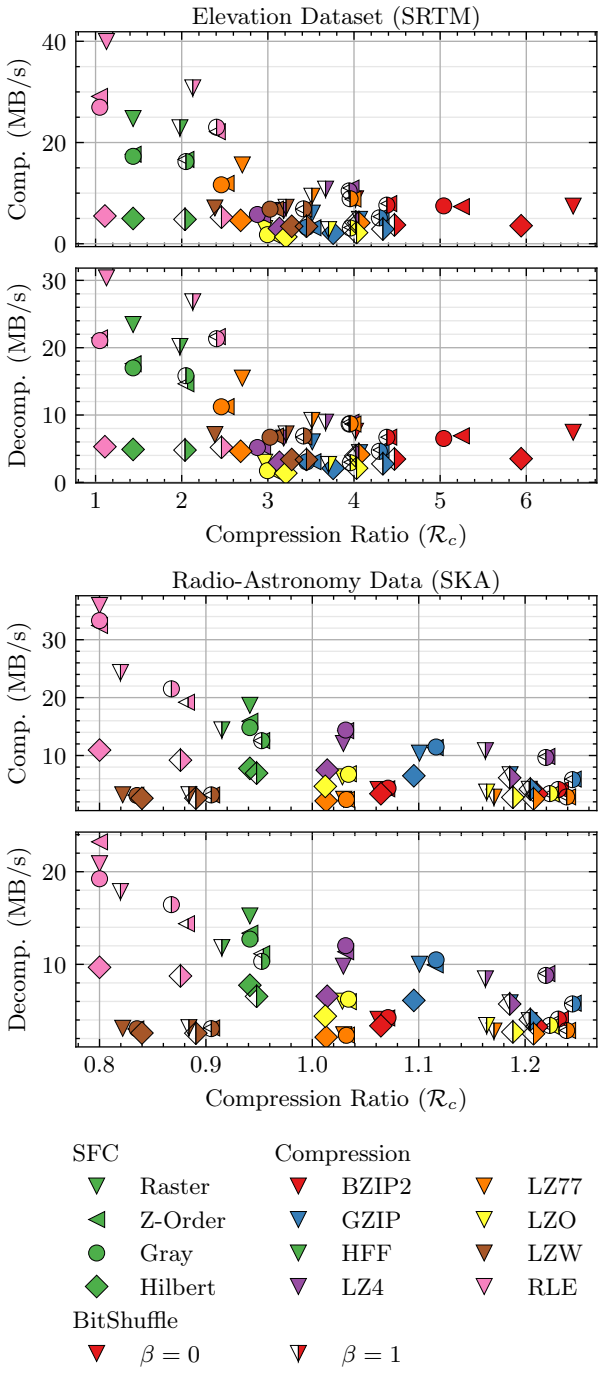


Figure 4.5: Compression and decompression speed versus compression ratio for all compression schemes, SFCs, and with and without BitShuffle. Note that the application of BitShuffle is denoted by $\beta = 1$.

for the lower speeds in the SKA dataset when BitShuffle is applied. However, the time elapsed to conduct BitShuffle is a few milliseconds at most and is independent of the compression scheme used. Therefore, any difference in compression and decompression speed between compression techniques, for a given dataset and β, must be because of the compression scheme and its interaction with BitShuffle and the SFCs, and not the BitShuffle implementation itself.

Though SFC reorderings do not positively impact compression speeds without BitShuffle for the SRTM dataset, the SKA dataset does benefit from their usage. When applying the Gray-Code Curve without BitShuffle to the SKA dataset, the compression and decompression speeds of LZ4 are increased by 19 % and 22.5 % respectively. This benefit is diminished when BitShuffle is applied, improving decompression speeds by under 10 % but slowing compression down by 10.6 %. However, the introduction of BitShuffle to any configuration for the SKA dataset nearly always reduces speeds. The only exceptions are LZ77 and LZW which are approximately 0.4 MB/s and 0.04 MB/s faster respectively. For all compression schemes and configurations, the Hilbert Curve is always the slowest. This can be partially attributed to the significantly slower mapping rate shown in Fig. 4.2. The total runtime for each compression scheme has a roughly-constant delay when using the Hilbert Curve compared to the Raster Scan and other two SFCs. For the SKA and SRTM datasets, this offset is 400 s to 500 s and approximately 1000 s respectively. The shorter delay when using the Hilbert Curve for the SKA dataset is caused by the approximately 1.5×10^5 Hz higher mapping rate for the three-dimensional Hilbert mapping with a sidelength of 2^7 compared to the two-dimensional Hilbert Mapping with a sidelength of 2^{11}.

For both datasets, the second fastest compression scheme is Huffman Encoding with the Raster Scan. This may be owing to the generation and processing of the Huffman tree but further research is required to identify the cause. Huffman Encoding is slowed down through the application of BitShuffle for both datasets, even with the Raster Scan. LZ77 is the third fastest scheme for the SRTM dataset but the slowest for the SKA dataset. This may be owing to the lack of an optimized dictionary search algorithm in the LZ77 implementation used, as the other LZ77-based schemes (GZIP, LZ4, and LZO) are faster. Of the compression schemes that achieve greater than

unity compression ratios for the SKA dataset, the fastest is LZ4 with BitShuffle and the Z-Order Curve. In fact, the Z-Order and Gray-Code Curves improve the compression and decompression speeds of BZIP2 and LZ4 for the SKA dataset, when compared to the Raster Scan.

When compared to the unpreprocessed configurations, the biggest percentage increase in speeds for the SRTM and SKA datasets are for LZ4 with the Z-Order Curve ($\beta = 1$) and the Gray-Code Curve ($\beta = 0$) respectively. This is also the case for the biggest absolute increase in compression and decompression speeds, ignoring RLE for the SKA dataset as it does not achieve a reduction in dataset size. With LZ4, BitShuffle, and the Z-Order Curve, the SRTM dataset is compressed at 11 MB/s and decompressed at 8.92 MB/s. With LZ4 and the Gray-Code Curve, the SKA dataset is compressed at 14.4 MB/s and decompressed at 12 MB/s. However, LZ4 only achieves a compression ratio of 1.03 for the SKA dataset without BitShuffle. With BitShuffle, the Z-Order Curve is better, with compression and decompression speeds of 9.84 MB/s and 9.01 MB/s respectively. This is slower than if the Raster Scan was used without BitShuffle. If BitShuffle is to be used on the SRTM dataset, the benefit of SFC reordering on the speed of LZ4 is lost, indicating that the majority of the speed improvement comes from the usage of BitShuffle and not SFCs. However, the speeds of GZIP and LZO ($\beta = 1$) are increased anywhere between 5 % and 15 % by applying the Z-Order or Gray-Code Curve, with the former introducing the biggest improvement.

The speed of LZ77 on the SKA dataset is increased by 15 % to 20 % by using BitShuffle, with no substantial improvement through the use of SFC reorderings. The accompanying speeds never reach 3 MB/s. Without BitShuffle, LZ77 is slower, more so when SFCs are applied. However, when LZ77 is used on the SRTM dataset, BitShuffle has the opposite effect, slowing down speeds with the Raster Scan always achieving the fastest compression and decompression. The significant improvement to compression ratios for LZW by using SFC reordering on the SRTM dataset does not translate to increases in speed. Compression and decompression speeds are 2 % to 6 % slower without BitShuffle and 4 % to 7 % slower with BitShuffle than when using the Raster Scan.

Within the same literature covered at the end of Section 4.2.2, only three explore compression speeds. The Hilbert Curve and DPCM-based compression scheme by Scarmana and McDougall has a compression speed of approximately 4 MB/s [12], which is around the middle for all compression schemes in Fig. 4.5 for the SRTM dataset. Their compression speeds for LZW and DEFLATE are comparable to those achieved in this work, though GZIP (DEFLATE) is slightly slower for the SRTM dataset. This may be because the results for GZIP in Table 4.1 and Fig. 4.5 use the highest compression level and thus take longer than with the default DEFLATE parameters from PNG, used by Scarmana and McDougall.

Comparing compression speeds for the SKA dataset, with the results obtained by Zheng *et al.* [58] and Masui *et al.* [20], is not appropriate as they use hardware configurations that differ greatly from that used here. Masui *et al.* execute their tests on data files stored entirely in memory, removing any I/O overhead of a permanent storage medium such as an SSD [20]. This is why they are able to achieve a write-speed of 749 MiB/s. Zheng *et al.* do not store their data files in memory, but they do use an FPGA as an external accelerator [58]. They do experiment with executing their technique on a CPU alone, but it is nearly seven times slower than BitShuffle with LZ4.

4.3 Discussion

SFC reordering has a significant impact on file-entropy and block-entropy for the SRTM dataset, without BitShuffle. Block-entropy values are on average 0.1 lower with SFCs than with the Raster Scan. When BitShuffle is applied, this impact is significantly reduced as BitShuffle is the primary reason for the reduction in entropy. However, the block-entropy for the most significant bits is effectively zero for at least half of the files when SFC orderings are used. The block-entropy with BitShuffle is about 0.15 lower with SFCs than with the Raster Scan. The high redundancy in the most significant bits of the SRTM data is caused by its high local-correlation and the integer representation used. The opposite effect is observed in the SKA dataset, with SFCs having a greater effect with BitShuffle than without. SFC reordering

has little to no effect on block-entropies for the SKA dataset when BitShuffle is not applied, with block-entropies just above 0.9. One reason for this is the low local-correlation in the SKA data, indicating that values are more dissimilar than alike within a small neighbouring region. The floating-point values are compact in their structure, but there are redundant sections within the binary representation. BitShuffle allows the components of the single-precision floating-point values with the smallest variation to be compressed: the exponent and sign bits. With BitShuffle and the Raster Scan, the block-entropy for the exponent and sign bit regions of the SKA dataset is around 0.4 lower than without BitShuffle. SFC reordering lowers this by a further 0.2, a massive reduction over the block-entropy without preprocessing. However, the mantissa makes up 72 % of each BitShuffled SKA file, resulting in file-entropies above 0.85 with SFC reordering and above 0.9 with the Raster Scan.

Of the three SFCs, the Hilbert Curve achieves the best compression ratios for the SRTM dataset without BitShuffle. However, the Raster Scan has the largest $\mathcal{R}_c$ for all compression schemes except for GZIP, LZO, and LZW. With BitShuffle applied to the SRTM dataset, the Raster Scan always has the smallest compression ratio while the Hilbert Curve has the largest; though it misses out by 0.01 to the Z-Order and Gray-Code Curves for Huffman Encoding. For the SKA dataset, the Hilbert Curve only achieves the largest compression ratio with LZW and no BitShuffle but lands up increasing the dataset size anyway ($\mathcal{R}_c = 0.84$). Regardless of whether BitShuffle is used, the Z-Order and Gray-Code Curves consistently achieve compression ratios larger than if the Raster Scan was used. Without BitShuffle, the largest $\mathcal{R}_c$ value is 1.12 with GZIP and either of the Z-Order or Gray-Code Curves. None of the other compression schemes achieve more than $\mathcal{R}_c = 1.1$ without BitShuffle. The top compressors for the SKA dataset — with BitShuffle — are BZIP2, GZIP, LZ4, LZ77, and LZO (all within compression ratios of $1.22 \leq \mathcal{R}_c \leq 1.25$). Without BitShuffle, the same compressors achieve maximum compression ratios of 1.07, 1.12, 1.03, 1.03, 1.03 respectively. The top compressors for the SRTM dataset are BZIP2 and GZIP with BitShuffle ($\mathcal{R}_c = 4.47$ and $\mathcal{R}_c = 4.33$) and without BitShuffle ($\mathcal{R}_c = 6.54$ and $\mathcal{R}_c = 3.76$).

Even though SFC reorderings have a generally positive impact on compression ratios, in only a few cases are the compression or decompression speeds also increased. The

time requirements of the Hilbert Curve algorithm and implementation used [117] results in impractical speed reductions, for both compression and decompression, across the board. Given that DEM data are not found in file-formats that support BitShuffle, the best compression configuration does appear to be BZIP2 with the Raster Scan and without BitShuffle: achieving a compression ratio of 6.54 and compression and decompression speeds of 7.43 MB/s. Of the compression schemes investigated, only Huffman Encoding, LZ77, LZW, and RLE are in the DEM file-formats identified in Table 2.2. Of these four schemes, LZW is improved the most by SFC reordering (Z-Order Curve), increasing the compression ratio from 2.39 to 3.09 while effectively maintaining compression and decompression speeds within a margin of 0.24 MB/s.

If a faster Hilbert Curve mapping implementation were to be integrated, a compression ratio of 3.28 could be obtained without the substantial speed penalty present in these results. Interestingly, GZIP is improved in all regards through the use of the Gray-Code Curve when applied to the SKA dataset without BitShuffle. The fastest scheme for the SRTM dataset is RLE followed by Huffman Encoding. In fact, RLE achieves faster compression and larger compression ratios than Huffman Encoding, when BitShuffle is used. The general lacklustre speeds with preprocessing are more than likely because of implementation specific details as is the case with the Hilbert Curve and BitShuffle on three-dimensional floating-point data. Further research in this regard is needed to determine how the speeds can be brought to be on par with the Raster Scan while maintaining the better compression ratios measured. If the Hilbert Curve and BitShuffle implementations are optimized for faster computation, then the compression ratio advantages they provide can be exploited without the speed penalty found in this work.

The SKA dataset is highly incompressible, as shown by the low compression ratios in Table 4.2. As is evident in the literature, BitShuffle has a significant impact on compression ratios, achieving a maximum of $\mathcal{R}_c = 1.25$ with GZIP and the Z-Order Curve compared to the maximum without BitShuffle of $\mathcal{R}_c = 1.12$ with the same scheme and SFC. Unlike for the SRTM data, the Hilbert Curve is never the best compressor for the SKA Dataset, with or without BitShuffle. Secondly, the Raster Scan is never the best compressor by itself, always tied with the Z-Order

and Gray-Code Curves. Furthermore, the Raster Scan is always the worst ordering for compression with BitShuffle for both datasets. This shows that the *best* SFC reordering scheme is different for these two types of datasets. The strong differences in their autocorrelations, coupled with their data types, is the clearest reason behind why the Hilbert Curve is best for the SRTM data and the Z-Order and Gray-Code Curves are best for the SKA data. As is shown by Mokbel, Aref, and Kamel, the Hilbert Curve has no bias to any given dimension whereas the Raster Scan, Z-Order, and Gray-Code curves have strong biases towards specific dimensions [36]. Given the SKA data's strong correlation in the time-dimension, compared to the channel and correlator dimensions, it is natural to ask whether a variant of the Raster Scan that traverses the dimensions in a different order would result in *better* compression performance than the orderings investigated here.

4.4 Conclusion from Experimental Results

The application of SFC reordering to DEM and radio-astronomy data gives varying results that are dependent more on the compression scheme than the curve used and whether BitShuffle is applied. SFCs are shown to reduce file-entropy and block-entropy for the SRTM dataset (without BitShuffle) and the SKA dataset (with BitShuffle). However, there is little difference between the Z-Order, Gray-Code, and Hilbert Curves, as they are all Recursive Space-Filling Curves with the same subquadrants and thus data extents. BitShuffle is shown to improve compression ratios across the board with varying speeds. The Z-Order and Gray-Code Curves improve compression ratios and maintain compression speeds for both datasets, when compared to the Raster Scan. However, the Hilbert Curve gives larger compression ratios for the SRTM dataset. Using the three curves, the compression ratio of the SRTM dataset using LZW is increased from 2.39 to 3.09 while maintaining a speed of around 7 MB/s using the Z-Order Curve.

GZIP and LZ4, combined with BitShuffle and the Z-Order Curve, achieve compression ratios of 1.24 and 1.22 for the SKA Dataset, with average speeds of 5.87 MB/s and 9.43 MB/s respectively. In fact, the compression performance of GZIP on the

SKA dataset, without BitShuffle, is improved in all regards by using the Gray-Code Curve ($\mathcal{R}_c$ from 1.1 to 1.2 and speeds from approximately 10 MB/s to approximately 11 MB/s). The Hilbert Curve consistently achieves higher compression ratios for the SRTM data with and without BitShuffle but loses to the Z-Order and Gray-Code Curves for the SKA dataset. However, compression speeds are significantly biased towards the standard Row-Major Order for the SRTM dataset, with the fastest configuration on the SKA dataset varying between Row-Major Order, the Z-Order Curve, and the Gray-Code Curve. Furthermore, the Hilbert Curve mapping implementation introduces a significant runtime penalty — compared to the Row-Major, Z-Order, and Gray-Code Curves — making its usage in these contexts impractical unless a faster mapping algorithm is identified and integrated.

The Z-Order and Gray-Code Curves show consistent improvement to compression ratios and varying improvement to processing speeds over the Raster Scan, for the SKA dataset with BitShuffle, indicating that alternative orderings do benefit compression performance for radio-astronomy data. If a faster Hilbert Mapping algorithm is identified, its use as a reordering scheme for DEM data shows significant promise for improving already existing compression schemes. Some additional speed improvements may be gained by further optimizing the BitShuffle implementation similarly to the Hilbert Curve implementation.

Chapter 5

Conclusion

5.1 Summary of Research

The work presented here centres around the problem of a lack of knowledge on the effect of SFC reorderings on compression performance for Digital Elevation Model and Radio-Astronomy data. Firstly, background information and a literature review is presented. Definitions for SFCs, locality preservation, and the measurement-aggregate-metric methodology are given, with a review of the applications of these curves to compression and otherwise. The two datasets used to answer the three research questions posed — the SRTM (DEM) and SKA (Radio-Astronomy) datasets — are analysed to identify the behaviour of their underlying values. A discussion of compression schemes and BitShuffle is also given.

The contributions made in this research are presented as two papers in Chapters 3 and 4. The second paper was published in the IEEE Access journal on the 28th of May 2021 [22]. The first paper (Chapter 3) contains an in-depth investigation into and comparison of metrics quantifying the locality preservation ability of SFCs. This addresses the first research question, which is given below. The second paper (Chapter 4) contains the core contribution of this work: the compression of DEM and Radio-Astronomy data. This includes a description of the methodology used and software implemented to carry out the compression process. Following this is a detailed analysis of the entropy and compression performance of the SFC reordered datasets.

5.2 Recommendations and Future Work

The compression of data mapped using SFCs is highly dependent on the behaviour of the data. For DEM data, it is recommended to use the Z-Order or Gray-Code Curve as it should give an adequate improvement to compression ratios without significantly reducing speeds. The Hilbert Curve can be used, but it will negatively impact compression and decompression speeds unless an optimized mapping algorithm is developed and utilized. Most Lempel-Ziv type compression schemes investigated had improvements to compression performance for DEM data through the use of SFC reorderings.

If BZIP2 is used, SFCs are not recommended as compression performance is not improved through their use. BitShuffle is absolutely necessary to effectively compress floating-point Radio-Astronomy data, with the Z-Order and Gray-Code Curves being the recommended SFCs. BZIP2 has no advantage over Lempel-Ziv type compression schemes for this type of data. As with all software and computing design decisions, changes to compression ratios and compression and decompression speeds must be balanced within the context of the application.

As discussed in Sections 4.2 and 4.3, the implementation of the Hilbert Curve mapping is the biggest opportunity for improvement as its usage improves compression ratios for the SRTM dataset but is significantly slower owing to the slow mapping rate. The mapping implementation for the Z-Order and Gray-Code Curves exploits AVX2 and BMI2 x86 instruction sets [116] whereas the Hilbert Curve implementation does not [118]. As is previously highlighted, the Z-Order and Gray-Code Curves perform best for the SKA dataset. However, given the stronger autocorrelation in the time-dimension, there is the possibility that a variant of the Raster Scan or another SFC would result in better compression performance if it can exploit this dimensional bias. The Description Vector metric, developed by Mokbel, Aref, and Grama, is an appropriate tool to evaluate the compression performance for a given SFC [36].

If further work is conducted on reordering radio-astronomy data for the purpose of compression, the description vector is an appropriate and necessary tool. The

compression ratios achieved are good, but comparing these results to those in the literature indicates that there may be other compression schemes that are better suited for the datasets with SFC reorderings [12], [20], [55], [58]. Applying the following methodology to other compression schemes would expand upon the knowledge of how such reorderings effect compression performance. The most interesting candidates, in the author's opinion, are arithmetic encoding, predictive coding (such as DPCM), and domain-specific lossy compression schemes. Furthermore, compressing floating-point DEM data using this methodology would help identify the impact of different data types on similar IGRF data-models.

5.3 Conclusion

Through the work presented here, the three research questions posed in Section 1.1 have been answered. In achieving this, a number of contributions are made, the core of which is in the application SFCs to the compression of two types of data that have not been extensively researched before: DEM and Radio-Astronomy data. The results from this contribution expand the knowledge of SFC-based compression methods in an extensive and exhaustive manner. Another significant contribution of this research is in the analysis and discussion of locality preservation metrics for SFCs. Conclusions made in this analysis are used in the former contribution, illustrating that an understanding of these metrics is important when applying SFCs to a computational problem. The answers to each of the three research questions originally posed are as follows.

What metrics can be used to rank SFCs for a specific computational task?
Numerous methods quantifying locality preservation were identified from the literature. In doing so, they were categorized into two types (point-pair and region metrics) and size subtypes (the Hölder Continuity Ratio, Reciprocal Hölder Continuity Ratio, Partition Ratio, Description and Irregularity Vectors, Bounding-Box Metrics, and Clustering Metric). It was shown that these metrics give rise to different *best* SFCs, which are also dependent on the size

of the SFC. Appropriate computational applications were identified by matching the algorithms, data structures, and metric-measurements. Through this analysis, the Description Vector metric is used to complement and analyse the compression performance achieved in Chapter 4.

What is the impact of SFC reorderings on the compressibility of DEM and Radio-Astronomy data? SFCs can improve compressibility by a respectable margin, but the degree of this improvement is based on the properties of the data. Utilizing DEM and Radio-Astronomy datasets, which present different spatial properties, it was shown that SFCs can improve compressibility when the curve traversal and the data correlations match. SFCs reduce block-entropy values; as was hypothesized. However, the data type (integer or floating-point) is a significant driving factor on the compression performance. BitShuffle, the bit-wise preprocessing step, allows SFCs to increase compressibility significantly more than without. The Z-Order, Gray-Code, and Hilbert Curves all improve the compressibility similarly.

What is the impact of SFC reorderings on compression performance for DEM and Radio-Astronomy data? Improvements to compression performance vary more than with compressibility. The Z-Order and Gray-Code Curves improve compression ratios for both datasets, but the Hilbert Curve only contributes such an improvement for the DEM data. This is attributed to the IGRF behaviour of the DEM data, a property that cannot be assigned to the SKA data. Though SFCs do contribute to larger compression ratios in some cases, the impact on compression and decompression speed is not as beneficial. Overall, SFCs do not reduce the execution time of compression. However, in many cases the speeds are maintained. The Hilbert Curve has a significant negative impact on compression and decompression speeds, owing to the higher computational cost and poor scaling of the Hilbert mapping algorithm compared to that of the Raster, Z-Order, and Gray-Code Curves. BitShuffle is necessary to effectively compress the SKA dataset, with SFCs providing an improvement to compression performance with and without it.

References

1. M. Bader, *Space-Filling Curves: An Introduction with Applications in Scientific Computing*, ser. Texts in Computational Science and Engineering 9. Heidelberg ; New York: Springer, 2013, 278 pp., ISBN: 978-3-642-31045-4.

2. H. Haverkort and F. van Walderveen, "Locality and bounding-box quality of two-dimensional space-filling curves," *Computational Geometry*, Special Issue on the 24th European Workshop on Computational Geometry (EuroCG'08), vol. 43, no. 2, pp. 131–147, Feb. 1, 2010, ISSN: 0925-7721. DOI: 10.1016/j.comgeo.2009.06.002. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0925772109000765.

3. J. K. Lawder and P. J. H. King, "Querying multi-dimensional data indexed using the Hilbert Space-filling Curve," *SIGMOD Rec.*, vol. 30, no. 1, pp. 19–24, Mar. 2001, ISSN: 0163-5808. DOI: 10.1145/373626.373678. [Online]. Available: http://doi.acm.org/10.1145/373626.373678.

4. D. Holzmüller, "Efficient neighbor-finding on space-filling curves," University of Stuttgart, Germany, Oct. 17, 2017. arXiv: 1710.06384. [Online]. Available: http://arxiv.org/abs/1710.06384.

5. L. Arge, M. D. Berg, H. Haverkort, and K. Yi, "The priority R-tree: A practically efficient and worst-case optimal R-tree," *ACM Transactions on Algorithms*, vol. 4, no. 1, 9:1–9:30, Mar. 28, 2008, ISSN: 1549-6325. DOI: 10.1145/1328911.1328920. [Online]. Available: https://doi.org/10.1145/1328911.1328920.

6. M. Bader and C. Zenger, "Cache oblivious matrix multiplication using an element ordering based on a Peano curve," *Linear Algebra and its Applications*, Special Issue in Honor of Friedrich Ludwig Bauer, vol. 417, no. 2, pp. 301–313, Sep. 1, 2006, ISSN: 0024-3795. DOI: 10.1016/j.laa.2006.03.018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0024379506001595.

7. M. Bader and A. Heinecke, "Cache oblivious dense and sparse matrix multiplication based on Peano curves," in *9th International Workshop on State-of-the-Art in Scientific and Parallel Computing*, vol. 9, Trondheim, Norway:

Springer, May 2008, p. 10. [Online]. Available: https://pdfs.semanticscholar.org/0ca3/471cced2f302d24d820f54e09b12ddba7eb5.pdf.

8. C. Bohm, M. Perdacher, and C. Plant, "A Novel Hilbert Curve for Cache-locality Preserving Loops," *IEEE Transactions on Big Data*, pp. 1–1, 2018, ISSN: 2332-7790. DOI: 10.1109/TBDATA.2018.2830378.

9. T. Bially, "Space-filling curves: Their generation and their application to bandwidth reduction," *IEEE Transactions on Information Theory*, vol. 15, no. 6, pp. 658–664, Nov. 1969, ISSN: 0018-9448. DOI: 10.1109/TIT.1969.1054385.

10. H. Lian, W. Qiu, D. Yan, Z. Huang, and P. Tang, "Efficient and secure k-nearest neighbor query on outsourced data," *Peer-to-Peer Networking and Applications*, vol. 13, no. 6, pp. 2324–2333, Nov. 1, 2020, ISSN: 1936-6450. DOI: 10.1007/s12083-020-00909-2. [Online]. Available: https://doi.org/10.1007/s12083-020-00909-2.

11. C. Faloutsos and S. Roseman, "Fractals for secondary key retrieval," in *Proceedings of the Eighth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems - PODS '89*, Philadelphia, Pennsylvania, United States: ACM Press, 1989, pp. 247–252, ISBN: 978-0-89791-308-9. DOI: 10.1145/73721.73746. [Online]. Available: http://portal.acm.org/citation.cfm?doid=73721.73746.

12. G. Scarmana and K. McDougall, "An application of space filling curves to digital terrain models," in *Proceedings of the XVI International Congress for Mine Surveying: Connecting Education and Industry*, Ellipsis Media, vol. 1, 2016, pp. 113–117.

13. F. Pinciroli, C. Combi, G. Pozzi, M. Negretto, L. Portoni, and G. Invernizzi, "A Peano-Hilbert derived algorithm for compression of angiocardiographic images," in *[1991] Proceedings Computers in Cardiology*, Sep. 1991, pp. 81–84. DOI: 10.1109/CIC.1991.169050.

14. J.-Y. Liang, C.-S. Chen, C.-H. Huang, and L. Liu, "Lossless compression of medical images using Hilbert space-filling curves," *Computerized Medical*

Imaging and Graphics, vol. 32, no. 3, pp. 174–182, Apr. 1, 2008, ISSN: 0895-6111. DOI: 10.1016/j.compmedimag.2007.11.002. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S089561110700167X.

15. A. Abdollahi, N. Bruce, S. Kamali, and R. Karim, "Lossless image compression using list update algorithms," in *String Processing and Information Retrieval*, N. R. Brisaboa and S. J. Puglisi, Eds., ser. Lecture Notes in Computer Science, Cham: Springer International Publishing, 2019, pp. 16–34, ISBN: 978-3-030-32686-9. DOI: 10.1007/978-3-030-32686-9_2.

16. L. Hawker, J. Rougier, J. Neal, P. Bates, L. Archer, and D. Yamazaki, "Implications of simulating global digital elevation models for flood inundation studies," *Water Resources Research*, vol. 54, no. 10, pp. 7910–7928, 2018, ISSN: 1944-7973. DOI: 10.1029/2018WR023279. [Online]. Available: https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2018WR023279.

17. K.-t. Chang, *Introduction to Geographic Information Systems*, 9th ed. New York, NY, USA: McGraw-Hill Education, 2019, ISBN: 978-1-259-92964-9. [Online]. Available: https://www.mheducation.com/highered/product/introduction-geographic-information-systems-chang/M9781259929649.html.

18. F. Boogaard, Z. Vojinovic, Y.-C. Chen, J. Kluck, and T.-P. Lin, "High Resolution Decision Maps for Urban Planning: A Combined Analysis of Urban Flooding and Thermal Stress Potential In Asia and Europe," *MATEC Web of Conferences*, vol. 103, p. 04 012, 2017, ISSN: 2261-236X. DOI: 10.1051/matecconf/201710304012. [Online]. Available: https://www.matec-conferences.org/articles/matecconf/abs/2017/17/matecconf_iscee2017_04012/matecconf_iscee2017_04012.html.

19. P. E. Dewdney, P. J. Hall, R. T. Schilizzi, and T. J. L. W. Lazio, "The Square Kilometre Array," *Proceedings of the IEEE*, vol. 97, no. 8, pp. 1482–1496, Aug. 2009, ISSN: 1558-2256. DOI: 10.1109/JPROC.2009.2021005. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/5136190/.

20. K. Masui, M. Amiri, L. Connor, M. Deng, M. Fandino, C. Höfer, M. Halpern, D. Hanna, A. D. Hincks, G. Hinshaw, J. M. Parra, L. B. Newburgh, J. R. Shaw, and K. Vanderlinde, "A compression scheme for radio data in high performance computing," *Astronomy and Computing*, vol. 12, pp. 181–190, Sep. 1, 2015,

ISSN: 2213-1337. DOI: 10.1016/j.ascom.2015.07.002. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2213133715000694.

21. The HDF Group, "Hierarchical Data Format, version 5," 1997–2019. [Online]. Available: http://www.hdfgroup.org/HDF5/.

22. C. J. Haupt, E. J. Otoo, and L. Cheng, "Effect of d-Dimensional Re-orderings on Lossless Compression of Radio-Astronomy and Digital Elevation Data," *IEEE Access*, vol. 9, pp. 80 415–80 433, 2021, ISSN: 2169-3536. DOI: 10.1109/ACCESS.2021.3084838. [Online]. Available: https://ieeexplore.ieee.org/document/9443202.

23. G. Peano, "Sur une courbe, qui remplit toute une aire plane," *Mathematische Annalen*, vol. 36, no. 1, pp. 157–160, Mar. 1, 1890, ISSN: 0025-5831, 1432-1807. DOI: 10.1007/BF01199438. [Online]. Available: https://link.springer.com/article/10.1007/BF01199438.

24. G. Cantor, "Ein beitrag zur mannigfaltigkeitslehre," *Journal für die reine und angewandte Mathematik*, vol. 1878, no. 84, pp. 242–258, 1878. DOI: 10.1515/crelle-1878-18788413. [Online]. Available: https://doi.org/10.1515/crelle-1878-18788413.

25. H. Sagan, *Space-Filling Curves*, ser. Universitext Series. Springer-Verlag, 1994, 193 pp., ISBN: 978-0-387-94265-0. [Online]. Available: https://books.google.co.za/books?id=oEFAAQAAIAAJ.

26. D. Hilbert, "Ueber die stetige Abbildung einer Line auf ein Flächenstück," *Mathematische Annalen*, vol. 38, no. 3, pp. 459–460, Sep. 1, 1891, ISSN: 0025-5831, 1432-1807. DOI: 10.1007/BF01199431. [Online]. Available: https://link.springer.com/article/10.1007/BF01199431.

27. H. L. Lebesgue, "Lecons sur l'integration et la recherche des fonctions primitives, professees au college de france (deuxiemeedition)," *Collection de monographies sur la theorie des fonctions, Paris*, 1928.

28. G. M. Morton, "A computer oriented geodetic data base and a new technique in file sequencing," 1966.

29. C. Faloutsos, "Multiattribute hashing using Gray Codes," in *Proceedings of the 1986 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '86, New York, NY, USA: ACM, 1986, pp. 227–238, ISBN: 978-0-89791-191-7. DOI: 10.1145/16894.16877. [Online]. Available: http://doi.acm.org/10.1145/16894.16877.

30. J.-m. Wierum, "Definition of a new circular space-filling curve $\beta\Omega$-indexing," Technical Report TR-001-03, 2002, p. 12. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.18.3487.

31. T. Asano, D. Ranjan, T. Roos, E. Welzl, and P. Widmayer, "Space-filling curves and their use in the design of geometric data structures," *Theoretical Computer Science*, vol. 181, no. 1, pp. 3–15, Jul. 15, 1997, ISSN: 0304-3975. DOI: 10.1016/S0304-3975(96)00259-9. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0304397596002599.

32. R. Niedermeier, K. Reinhardt, and P. Sanders, "Towards optimal locality in mesh-indexings," *Discrete Applied Mathematics*, vol. 117, no. 1, pp. 211–237, Mar. 15, 2002, ISSN: 0166-218X. DOI: 10.1016/S0166-218X(00)00326-7. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0166218X00003267.

33. P. Xu, C. Nguyen, and S. Tirthapura, "Onion Curve: A space filling curve with near-optimal clustering," in *2018 IEEE 34th International Conference on Data Engineering (ICDE)*, Jan. 23, 2018, pp. 1236–1239. DOI: 10.1109/ICDE.2018.00119.

34. C. Gotsman and M. Lindenbaum, "On the metric properties of discrete space-filling curves," in *Proceedings of the 12th IAPR International Conference on Pattern Recognition, Vol. 2 - Conference B: Computer Vision Image Processing. (Cat. No.94CH3440-5)*, Oct. 1994, 98–102 vol.3. DOI: 10.1109/ICPR.1994.577130.

35. B. Moon, H. V. Jagadish, C. Faloutsos, and J. H. Saltz, "Analysis of the clustering properties of the Hilbert space-filling curve," *IEEE Transactions on Knowledge and Data Engineering*, vol. 13, no. 1, pp. 124–141, Jan. 2001, ISSN: 1041-4347. DOI: 10.1109/69.908985.

36. M. F. Mokbel, W. G. Aref, and I. Kamel, "Analysis of multi-dimensional space-filling curves," *GeoInformatica*, vol. 7, no. 3, pp. 179–209, Sep. 1, 2003, ISSN: 1384-6175, 1573-7624. DOI: 10.1023/A:1025196714293. [Online]. Available: https://link.springer.com/article/10.1023/A:1025196714293.

37. F. Gray, "Pulse code communication," U.S. Patent 2632058A, Mar. 17, 1953. [Online]. Available: https://patents.google.com/patent/US2632058A/en.

38. E. Netto, "Beitrag zur Mannigfaltigkeitslehre.," *Journal für die reine und angewandte Mathematik*, vol. 1879, no. 86, pp. 263–268, 1879. DOI: 10.1515/crll.1879.86.263. [Online]. Available: https://doi.org/10.1515/crll.1879.86.263.

39. A. Lempel and J. Ziv, "Compression of two-dimensional data," *IEEE Transactions on Information Theory*, vol. 32, no. 1, pp. 2–8, Jan. 1986, ISSN: 0018-9448. DOI: 10.1109/TIT.1986.1057132.

40. P. Xu and S. Tirthapura, "On the optimality of clustering properties of space filling curves," in *Proceedings of the 31st ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*, ser. PODS '12, Scottsdale, Arizona, USA: Association for Computing Machinery, May 21, 2012, pp. 215–224, ISBN: 978-1-4503-1248-6. DOI: 10.1145/2213556.2213587. [Online]. Available: https://doi.org/10.1145/2213556.2213587.

41. J. Ziv and A. Lempel, "A universal algorithm for sequential data compression," *IEEE Transactions on Information Theory*, vol. 23, no. 3, pp. 337–343, May 1977, ISSN: 0018-9448. DOI: 10.1109/TIT.1977.1055714.

42. J. Ziv and A. Lempel, "Compression of individual sequences via variable-rate coding," *IEEE Transactions on Information Theory*, vol. 24, no. 5, pp. 530–536, Sep. 1978, ISSN: 0018-9448. DOI: 10.1109/TIT.1978.1055934.

43. F. J. Merkl, "Binary image compression using run length encoding and multiple scanning techniques," M.Sc. Thesis, Rochester Institute of Technology, Rochester New York, May 16, 1988, 64 pp. [Online]. Available: https://scholarworks.rit.edu/theses/184.

44. Welch, "A technique for high-performance data compression," *Computer*, vol. 17, no. 6, pp. 8–19, Jun. 1984, ISSN: 0018-9162. DOI: 10.1109/MC.1984.1659158.

45. D. A. Huffman, "A method for the construction of minimum-redundancy codes," *Proceedings of the IRE*, vol. 40, no. 9, pp. 1098–1101, Sep. 1952, ISSN: 0096-8390. DOI: 10.1109/JRPROC.1952.273898.

46. J. A. Provine and R. M. Rangayyan, "Lossless compression of Peanoscanned images," *Journal of Electronic Imaging*, vol. 3, no. 2, pp. 176–181, Apr. 1994, ISSN: 1017-9909, 1560-229X. DOI: 10.1117/12.171928. [Online]. Available: https://www.spiedigitallibrary.org/journals/journal-of-electronic-imaging/volume-3/issue-2/0000/Lossless-compression-of-Peanoscanned-images/10.1117/12.171928.short.

47. T. Ouni, A. Lassoued, and M. Abid, "Gradient-based Space Filling Curves: Application to lossless image compression," in *2011 IEEE International Conference on Computer Applications and Industrial Electronics (ICCAIE)*, Dec. 2011, pp. 437–442. DOI: 10.1109/ICCAIE.2011.6162175.

48. T. Ouni, A. Lassoued, and M. Abid, "Lossless image compression using gradient based space filling curves (G-SFC)," *Signal, Image and Video Processing*, vol. 9, no. 2, pp. 277–293, Feb. 1, 2015, ISSN: 1863-1711. DOI: 10.1007/s11760-013-0435-4. [Online]. Available: https://doi.org/10.1007/s11760-013-0435-4.

49. CompuServe Incorporated, *Graphics Interchange Format (89a): Programming Reference*, Jul. 31, 1990. [Online]. Available: https://www.w3.org/Graphics/GIF/spec-gif89a.txt.

50. T. Boutell, "PNG (Portable Network Graphics) specification version 1.0," 1997, ISSN: 2070-1721. [Online]. Available: https://www.rfc-editor.org/info/rfc2083.

51. P. Deutsch, "DEFLATE compressed data format specification version 1.3," RFC Editor, RFC RFC1951, May 1996. DOI: 10.17487/RFC1951. [Online]. Available: https://www.rfc-editor.org/info/rfc1951.

52. R. Pračko and J. Polec, "DPCM application to images scanned by SFC method," *Journal of ELECTRICAL ENGINEERING*, vol. 58, no. 3, p. 4, Nov. 3, 2007.

53. A. Quin and Y. Yanagisawa, "On data compaction of scanning curves," *The Computer Journal*, vol. 32, no. 6, pp. 563–566, Dec. 1, 1989, ISSN: 0010-4620.

DOI: 10.1093/comjnl/32.6.563. [Online]. Available: https://academic.oup.com/comjnl/article/32/6/563/341695.

54. N. Memon, D. L. Neuhoff, and S. Shende, "An analysis of some common scanning techniques for lossless image coding," *IEEE Transactions on Image Processing*, vol. 9, no. 11, pp. 1837–1848, Nov. 2000, ISSN: 1057-7149. DOI: 10.1109/83.877207.

55. S. D. Rane and G. Sapiro, "Evaluation of JPEG-LS, the new lossless and controlled-lossy still image compression standard, for compression of high-resolution elevation data," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 39, no. 10, pp. 2298–2306, Oct. 2001, ISSN: 0196-2892. DOI: 10.1109/36.957293.

56. Y. Collet, *LZ4*, version 1.9.2, lz4, Jan. 13, 2020. [Online]. Available: https://github.com/lz4/lz4.

57. R. Nan, D. Li, C. Jin, Q. Wang, L. Zhu, W. Zhu, H. Zhang, Y. Yue, and L. Qian, "The five-hundred-meter aperture spherical radio telescope (FAST) project," *International Journal of Modern Physics D*, vol. 20, no. 06, pp. 989–1024, Jun. 5, 2011, ISSN: 0218-2718. DOI: 10.1142/S0218271811019335. [Online]. Available: https://www.worldscientific.com/doi/abs/10.1142/S0218271811019335.

58. Y. Zheng, Y. Zhu, Y. Song, T. Nan, and W. Li, "A Lossless Astronomical Data Compression Scheme with FPGA Acceleration," in *2019 32nd IEEE International System-on-Chip Conference (SOCC)*, Sep. 2019, pp. 45–49. DOI: 10.1109/SOCC46988.2019.1570562786.

59. H. V. Jagadish, "Linear clustering of objects with multiple attributes," in *Proceedings of the 1990 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '90, New York, NY, USA: ACM, 1990, pp. 332–342, ISBN: 978-0-89791-365-2. DOI: 10.1145/93597.98742. [Online]. Available: http://doi.acm.org/10.1145/93597.98742.

60. OSGeo Project, *PostGIS 3.0.1dev manual*, Feb. 8, 2020. [Online]. Available: https://postgis.net/stuff/postgis-3.0.pdf.

61. J. Qi, G. Liu, C. S. Jensen, and L. Kulik, "Effectively learning spatial indices," *Proceedings of the VLDB Endowment*, vol. 13, no. 12, pp. 2341–2354, Jul. 1, 2020, ISSN: 2150-8097. DOI: 10.14778/3407790.3407829. [Online]. Available: https://dl.acm.org/doi/10.14778/3407790.3407829.

62. H. Haverkort. "An inventory of three-dimensional Hilbert space-filling curves." arXiv: 1109.2323 [cs]. (Sep. 11, 2011), [Online]. Available: http://arxiv.org/abs/1109.2323 (visited on 06/13/2018).

63. S. Sieranoja and P. Fränti, "Constructing a High-Dimensional kNN-Graph Using a Z-Order Curve," *ACM Journal of Experimental Algorithmics*, vol. 23, 1.9:1–1.9:21, Oct. 22, 2018, ISSN: 1084-6654. DOI: 10.1145/3274656. [Online]. Available: https://doi.org/10.1145/3274656.

64. J. Qi, Y. Tao, Y. Chang, and R. Zhang, "Packing R-trees with Space-filling Curves: Theoretical Optimality, Empirical Efficiency, and Bulk-loading Parallelizability," *ACM Transactions on Database Systems*, vol. 45, no. 3, pp. 1–47, Sep. 25, 2020, ISSN: 0362-5915, 1557-4644. DOI: 10.1145/3397506. [Online]. Available: https://dl.acm.org/doi/10.1145/3397506.

65. U. Sakoglu, L. Bhupati, N. Beheshti, N. Tsekos, and L. Johnsson, "An Adaptive Space-Filling Curve Trajectory for Ordering 3D Datasets to 1D: Application to Brain Magnetic Resonance Imaging Data for Classification," in *Computational Science – ICCS 2020*, V. V. Krzhizhanovskaya, G. Závodszky, M. H. Lees, J. J. Dongarra, P. M. A. Sloot, S. Brissos, and J. Teixeira, Eds., ser. Lecture Notes in Computer Science, Cham: Springer International Publishing, 2020, pp. 635–646, ISBN: 978-3-030-50420-5. DOI: 10.1007/978-3-030-50420-5_48.

66. P. Tsinganos, B. Cornelis, J. Cornelis, B. Jansen, and A. Skodras, "A Hilbert Curve Based Representation of sEMG Signals for Gesture Recognition," in *2019 International Conference on Systems, Signals and Image Processing (IWSSIP)*, Jun. 2019, pp. 201–206. DOI: 10.1109/IWSSIP.2019.8787290.

67. P. Tsinganos, B. Cornelis, J. Cornelis, B. Jansen, and A. Skodras, "Hilbert sEMG data scanning for hand gesture recognition based on deep learning," *Neural Computing and Applications*, vol. 33, no. 7, pp. 2645–2666, Apr. 1, 2021, ISSN: 1433-3058. DOI: 10.1007/s00521-020-05128-7. [Online]. Available: https://doi.org/10.1007/s00521-020-05128-7.

68. Z. Ren, G. Chen, and W. Lu, "Space filling curve mapping for malware detection and classification," in *Proceedings of the 2020 3rd International Conference on Computer Science and Software Engineering*, ser. CSSE 2020, New York, NY, USA: Association for Computing Machinery, May 22, 2020, pp. 176–180, ISBN: 978-1-4503-7552-8. DOI: 10.1145/3403746.3403924. [Online]. Available: https://doi.org/10.1145/3403746.3403924.

69. N.-H. Lee, "Euclidean Plane," in *Geometry: From Isometries to Special Relativity*, ser. Undergraduate Texts in Mathematics. Cham: Springer International Publishing, 2020, pp. 1–22, ISBN: 978-3-030-42100-7. DOI: 10.1007/978-3-030-42101-4_1. [Online]. Available: http://link.springer.com/10.1007/978-3-030-42101-4_1.

70. T. G. Farr, P. A. Rosen, E. Caro, R. Crippen, R. Duren, S. Hensley, M. Kobrick, M. Paller, E. Rodriguez, L. Roth, D. Seal, S. Shaffer, J. Shimada, J. Umland, M. Werner, M. Oskin, D. Burbank, and D. Alsdorf, "The Shuttle Radar Topography Mission," *Reviews of Geophysics*, vol. 45, no. 2, RG2004, May 19, 2007, ISSN: 8755-1209. DOI: 10.1029/2005RG000183. [Online]. Available: http://doi.wiley.com/10.1029/2005RG000183.

71. USGS. "USGS EarthExplorer," Earth Explorer. (2018), [Online]. Available: https://earthexplorer.usgs.gov/ (visited on 11/06/2017).

72. F. J. Aguilar, M. A. Aguilar, J. L. Blanco, A. Nemmaoui, and A. M. García Lorca, "Analysis and validation of grid DEM generation based on Gaussian Markov random field," *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. XLI-B2, pp. 277–284, 2016. DOI: 10.5194/isprs-archives-XLI-B2-277-2016. [Online]. Available: https://www.int-arch-photogramm-remote-sens-spatial-inf-sci.net/XLI-B2/277/2016/.

73. SARAO, *SARAO Archive*, version 1.1.0, Feb. 2020. [Online]. Available: https://archive.sarao.ac.za/.

74. I. Sihlangu, "The MeerKAT Radio Frequency Interference Environment," 2019. [Online]. Available: https://open.uct.ac.za/handle/11427/31748.

75. Microprocessor Standards Committee, "IEEE standard for floating-point arithmetic," *IEEE Std 754-2019 (Revision of IEEE 754-2008)*, pp. 1–84, Jul. 2019. DOI: 10.1109/IEEESTD.2019.8766229.

76. Open Geospatial Consortium, *OGC GeoTIFF Standard*, Sep. 14, 2019. [Online]. Available: http://www.opengis.net/doc/IS/GeoTIFF/1.1.

77. W. Büscher. "Spectrum Lab manual - spectrum displays," DL4YHF's Ham Radio Homepage. (Oct. 23, 2020), [Online]. Available: https://www.qsl.net/dl4yhf/speclab/specdisp.htm#waterfall.

78. J. R. Seward, *BZIP2 and libBZIP2*, Jan. 26, 2002. [Online]. Available: https://www.sourceware.org/bzip2/.

79. J. Tsai, "BZIP2: Format specification," Mar. 17, 2016. [Online]. Available: https://github.com/dsnet/compress/blob/master/doc/bzip2-format.pdf.

80. Canonical Ltd, *Ubuntu – Software Packages in "bionic", Subsection utils*, 2020. [Online]. Available: https://packages.ubuntu.com/bionic/utils/.

81. M. F. Oberhumer, *LZO-a real-time data compression library*, version 2.1.0, 2008. [Online]. Available: https://www.oberhumer.com/opensource/lzo/.

82. G. Lampert, *Compression agorithms*, Nov. 26, 2020. [Online]. Available: https://github.com/glampert/compression-algorithms.

83. I. Tkatchev, *Yet another LZ77 (yalz77)*, Jun. 4, 2020. [Online]. Available: https://github.com/ivan-tkatchev/yalz77.

84. The HDF Group. "HDF5 filters," HDF Support Portal. (2006), [Online]. Available: https://portal.hdfgroup.org/display/support/Filters (visited on 02/29/2020).

85. R. Fiorenza, "Hölder and locally hölder continuous functions. the linear spaces $C^k(\Omega), C^{k,\lambda}(\Omega),$ and $C^{k,\lambda}_{\text{loc}}(\Omega)$," in *Hölder and locally Hölder Continuous Functions, and Open Sets of Class $C^k, C^{k,lambda}$*, R. Fiorenza, Ed., Cham: Springer International Publishing, 2016, pp. 1–35, ISBN: 978-3-319-47940-8. DOI: 10.1007/978-3-319-47940-8_1. [Online]. Available: https://doi.org/10.1007/978-3-319-47940-8_1.

86. M. Bader, "Locality properties of space-filling curves," in *Space-Filling Curves: An Introduction with Applications in Scientific Computing*, ser. Texts in Computational Science and Engineering, M. Bader, Ed., Berlin, Heidelberg: Springer, 2013, pp. 167–180, ISBN: 978-3-642-31046-1. DOI: 10.1007/978-3-642-31046-1_11. [Online]. Available: https://doi.org/10.1007/978-3-642-31046-1_11.

87. C. Gotsman and M. Lindenbaum, "On the metric properties of discrete space-filling curves," *IEEE Transactions on Image Processing*, vol. 5, no. 5, pp. 794–797, May 1996, ISSN: 1941-0042. DOI: 10.1109/83.499920.

88. G. Chochia, M. Cole, and T. Heywood, "Implementing the hierarchical PRAM on the 2D mesh: Analyses and experiments," in *Proceedings.Seventh IEEE Symposium on Parallel and Distributed Processing*, Oct. 1995, pp. 587–594. DOI: 10.1109/SPDP.1995.530736.

89. J. Alber and R. Niedermeier, "On multidimensional curves with Hilbert property," *Theory of Computing Systems*, vol. 33, no. 4, pp. 295–312, Aug. 1, 2000, ISSN: 1433-0490. DOI: 10.1007/s002240010003. [Online]. Available: https://doi.org/10.1007/s002240010003.

90. K. E. Bauman, "The dilation factor of the Peano-Hilbert curve," *Mathematical Notes*, vol. 80, no. 5-6, pp. 609–620, Nov. 2006, ISSN: 0001-4346, 1573-8876. DOI: 10.1007/s11006-006-0182-8. [Online]. Available: http://link.springer.com/10.1007/s11006-006-0182-8.

91. H. K. Dai and H. C. Su, "Norm-based locality measures of two-dimensional Hilbert curves," in *Algorithmic Aspects in Information and Management*, R. Dondi, G. Fertin, and G. Mauri, Eds., ser. Lecture Notes in Computer Science, Cham: Springer International Publishing, 2016, pp. 14–25, ISBN: 978-3-319-41168-2. DOI: 10.1007/978-3-319-41168-2_2.

92. A. A. Korneev and E. V. Shchepin, "L∞-locality of three-dimensional Peano curves," *Proceedings of the Steklov Institute of Mathematics*, vol. 302, no. 1, pp. 217–249, Aug. 1, 2018, ISSN: 1531-8605. DOI: 10.1134/S0081543818060111. [Online]. Available: https://doi.org/10.1134/S0081543818060111.

93. H. K. Dai and H. C. Su, "On p-norm based locality measures of space-filling curves," in *Algorithms and Computation*, R. Fleischer and G. Trippen, Eds., ser. Lecture Notes in Computer Science, Berlin, Heidelberg: Springer, 2005, pp. 364–376, ISBN: 978-3-540-30551-4. DOI: 10.1007/978-3-540-30551-4_33.

94. H. Haverkort, "How many three-dimensional Hilbert curves are there?" *Journal of Computational Geometry*, vol. 8, no. 1, pp. 206–281-206–281, 1 Sep. 12, 2017, ISSN: 1920-180X. [Online]. Available: https://jocg.org/index.php/jocg/article/view/3036.

95. J. Hungershöfer and J.-M. Wierum, "On the quality of partitions based on space-filling curves," in *Computational Science — ICCS 2002*, P. M. A. Sloot, A. G. Hoekstra, C. J. K. Tan, and J. J. Dongarra, Eds., ser. Lecture Notes in Computer Science, Berlin, Heidelberg: Springer, 2002, pp. 36–45, ISBN: 978-3-540-47789-1. DOI: 10.1007/3-540-47789-6_4.

96. G. W. Zumbusch, "On the quality of space-filling curve induced partitions," *ZAMM Journal of Applied Mathematics and Mechanics*, vol. 81, pp. 25–28, 2001. [Online]. Available: https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.32.9927.

97. A. Perez, S. Kamata, and E. Kawaguchi, "Peano scanning of arbitrary size images," in *Proceedings., 11th IAPR International Conference on Pattern Recognition. Vol. III. Conference C: Image, Speech and Signal Analysis,*, Aug. 1992, pp. 565–568. DOI: 10.1109/ICPR.1992.202050.

98. M. Pérez, X. Benavent, and R. Olanda, "Efficient coding of Quadtree nodes," in *Computational Science – ICCS 2006*, V. N. Alexandrov, G. D. van Albada, P. M. A. Sloot, and J. Dongarra, Eds., ser. Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2006, pp. 13–16, ISBN: 978-3-540-34384-4.

99. L. H. Harper, "Stabilization and the edgesum problem," *Ars Combinatoria*, vol. 4, pp. 225–270, 1977, ISSN: 0381-7032.

100. G. Mitchison and R. Durbin, "Optimal numberings of an NxN array," *SIAM Journal on Algebraic Discrete Methods*, vol. 7, no. 4, pp. 571–582, Oct. 1, 1986, ISSN: 0196-5212. DOI: 10.1137/0607063. [Online]. Available: https://epubs.siam.org/doi/abs/10.1137/0607063.

101. H. K. Dai and H. C. Su, "On the locality properties of space-filling curves," in *Algorithms and Computation*, T. Ibaraki, N. Katoh, and H. Ono, Eds., ser. Lecture Notes in Computer Science, Berlin, Heidelberg: Springer, 2003, pp. 385–394, ISBN: 978-3-540-24587-2. DOI: 10.1007/978-3-540-24587-2_40.

102. P. Fishburn, P. Tetali, and P. Winkler, "Optimal linear arrangement of a rectangular grid," *Discrete Mathematics*, vol. 213, no. 1-3, pp. 123–139, Feb. 2000, ISSN: 0012365X. DOI: 10.1016/S0012-365X(99)00173-9. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0012365X99001739.

103. P. Xu and S. Tirthapura, "A lower bound on proximity preservation by space filling curves," in *2012 IEEE 26th International Parallel and Distributed Processing Symposium*, May 2012, pp. 1295–1305. DOI: 10.1109/IPDPS.2012.118.

104. J.-m. Wierum, "Logarithmic path-length in space-filling curves," presented at the Canadian Conference on Computational GEometry, 2002, pp. 22–26.

105. M. F. Mokbel, W. G. Aref, and I. Kamel, "Performance of Multi-dimensional Space-filling Curves," in *Proceedings of the 10th ACM International Symposium on Advances in Geographic Information Systems*, ser. GIS '02, New York, NY, USA: ACM, 2002, pp. 149–154, ISBN: 978-1-58113-591-6. DOI: 10.1145/585147.585179. [Online]. Available: http://doi.acm.org/10.1145/585147.585179.

106. M. F. Mokbel and W. G. Aref, "Irregularity in multi-dimensional space-filling curves with applications in multimedia databases," in *Proceedings of the Tenth International Conference on Information and Knowledge Management*, ser. CIKM '01, New York, NY, USA: ACM, 2001, pp. 512–519, ISBN: 978-1-58113-436-0. DOI: 10.1145/502585.502671. [Online]. Available: http://doi.acm.org/10.1145/502585.502671.

107. M. F. Mokbel and W. G. Aref, "Irregularity in high-dimensional space-filling curves," *Distributed and Parallel Databases*, vol. 29, no. 3, pp. 217–238, Jun. 1, 2011, ISSN: 1573-7578. DOI: 10.1007/s10619-010-7070-7. [Online]. Available: https://doi.org/10.1007/s10619-010-7070-7.

108. H. V. Jagadish, "Analysis of the Hilbert curve for representing two-dimensional space," *Information Processing Letters*, vol. 62, no. 1, pp. 17–22, Apr. 14, 1997,

ISSN: 0020-0190. DOI: 10.1016/S0020-0190(97)00014-8. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0020019097000148.

109. J. A. Orenstein, "Redundancy in spatial databases," in *Proceedings of the 1989 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '89, New York, NY, USA: ACM, 1989, pp. 295–305, ISBN: 978-0-89791-317-1. DOI: 10.1145/67544.66954. [Online]. Available: http://doi.acm.org/10.1145/67544.66954.

110. D. Deford and A. Kalyanaraman, "Empirical Analysis of Space-Filling Curves for Scientific Computing Applications," in *2013 42nd International Conference on Parallel Processing*, Oct. 2013, pp. 170–179. DOI: 10.1109/ICPP.2013.26.

111. W. Hua and C. Zhou, "Clusters and filling-curve-based storage assignment in a circuit board assembly kitting area," *IIE Transactions*, vol. 40, no. 6, pp. 569–585, Apr. 11, 2008, ISSN: 0740-817X. DOI: 10.1080/07408170701503462. [Online]. Available: https://doi.org/10.1080/07408170701503462.

112. S. N. Bhatt and F. Thomson Leighton, "A framework for solving VLSI graph layout problems," *Journal of Computer and System Sciences*, vol. 28, no. 2, pp. 300–343, Apr. 1, 1984, ISSN: 0022-0000. DOI: 10.1016/0022-0000(84)90071-0. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0022000084900710.

113. A. Guttman, "R-trees: A dynamic index structure for spatial searching," in *Proceedings of the 1984 ACM SIGMOD International Conference on Management of Data*, (Boston, Massachusetts), ser. SIGMOD '84, New York, NY, USA: ACM, 1984, pp. 47–57, ISBN: 978-0-89791-128-3. DOI: 10.1145/602259.602266. [Online]. Available: http://doi.acm.org/10.1145/602259.602266.

114. R. A. Finkel and J. L. Bentley, "Quad trees a data structure for retrieval on composite keys," *Acta Informatica*, vol. 4, no. 1, pp. 1–9, Mar. 1, 1974, ISSN: 1432-0525. DOI: 10.1007/BF00288933. [Online]. Available: https://doi.org/10.1007/BF00288933.

115. O. Tange, "GNU Parallel - the command-line power tool," *;login: The USENIX Magazine*, vol. 36, no. 1, pp. 42–47, Feb. 2011. [Online]. Available: http://www.gnu.org/s/parallel.

116. J. Baert, *Libmorton: C++ Morton Encoding/Decoding library*, 2018. [Online]. Available: https://github.com/Forceflow/libmorton.

117. L. Chenyang, Z. Hong, and W. Nengchao, "Fast N-dimensional Hilbert mapping algorithm," in *2008 International Conference on Computational Sciences and Its Applications*, Jun. 2008, pp. 507–513. DOI: 10.1109/ICCSA.2008.74.

118. A. M. Partl, *Library for N-dimensional Hilbert mapping*, AIP E-Science, 2013. [Online]. Available: https://github.com/aipescience/libhilbert.

119. P. Deutsch, "GZIP file format specification version 4.3," RFC Editor, RFC1952, May 1996. DOI: 10.17487/rfc1952. [Online]. Available: https://www.rfc-editor.org/info/rfc1952.

120. M. F. Mokbel, W. G. Aref, and A. Grama, "Spectral LPM: An optimal locality-preserving mapping using the spectral (not fractal) order," in *Proceedings 19th International Conference on Data Engineering (Cat. No.03CH37405)*, Mar. 2003, pp. 699–701. DOI: 10.1109/ICDE.2003.1260840.