The Consultative Committee for Space Data Systems

Report Concerning Space Data System Standards

LOSSLESS DATA COMPRESSION

INFORMATIONAL REPORT

CCSDS 120.0-G-4

GREEN BOOK

November 2021

The Consultative Committee for Space Data Systems

Report Concerning Space Data System Standards

LOSSLESS DATA COMPRESSION

INFORMATIONAL REPORT

CCSDS 120.0-G-4

GREEN BOOK

November 2021

AUTHORITY

Issue:	Informational Report, Issue 4
Date:	November 2021
Location:	Washington, DC, USA

This document has been approved for publication by the Management Council of the Consultative Committee for Space Data Systems (CCSDS) and reflects the consensus of technical panel experts from CCSDS Member Agencies. The procedure for review and authorization of CCSDS Reports is detailed in *Organization and Processes for the Consultative Committee for Space Data Systems* (CCSDS A02.1-Y-4).

This document is published and maintained by:

CCSDS Secretariat
National Aeronautics and Space Administration
Washington, DC, USA
Email: secretariat@mailman.ccsds.org

FOREWORD

This document is a CCSDS Report which contains background and explanatory material to support the CCSDS Recommendation, *Lossless Data Compression* (reference [1]).

Through the process of normal evolution, it is expected that expansion, deletion, or modification of this document may occur. This Report is therefore subject to CCSDS document management and change control procedures, which are defined in *Organization and Processes for the Consultative Committee for Space Data Systems* (CCSDS A02.1-Y-4). Current versions of CCSDS documents are maintained at the CCSDS Web site:

http://www.ccsds.org/

Questions relating to the contents or status of this document should be sent to the CCSDS Secretariat at the email address indicated on page i.

At time of publication, the active Member and Observer Agencies of the CCSDS were:

Member Agencies

- Agenzia Spaziale Italiana (ASI)/Italy.
- Canadian Space Agency (CSA)/Canada.
- Centre National d'Etudes Spatiales (CNES)/France.
- China National Space Administration (CNSA)/People's Republic of China.
- Deutsches Zentrum für Luft- und Raumfahrt (DLR)/Germany.
- European Space Agency (ESA)/Europe.
- Federal Space Agency (FSA)/Russian Federation.
- Instituto Nacional de Pesquisas Espaciais (INPE)/Brazil.
- Japan Aerospace Exploration Agency (JAXA)/Japan.
- National Aeronautics and Space Administration (NASA)/USA.
- UK Space Agency/United Kingdom.

Observer Agencies

- Austrian Space Agency (ASA)/Austria.
- Belgian Science Policy Office (BELSPO)/Belgium.
- Central Research Institute of Machine Building (TsNIIMash)/Russian Federation.
- China Satellite Launch and Tracking Control General, Beijing Institute of Tracking and Telecommunications Technology (CLTC/BITTT)/China.
- Chinese Academy of Sciences (CAS)/China.
- China Academy of Space Technology (CAST)/China.
- Commonwealth Scientific and Industrial Research Organization (CSIRO)/Australia.
- Danish National Space Center (DNSC)/Denmark.
- Departamento de Ciência e Tecnologia Aeroespacial (DCTA)/Brazil.
- Electronics and Telecommunications Research Institute (ETRI)/Korea.
- European Organization for the Exploitation of Meteorological Satellites (EUMETSAT)/Europe.
- European Telecommunications Satellite Organization (EUTELSAT)/Europe.
- Geo-Informatics and Space Technology Development Agency (GISTDA)/Thailand.
- Hellenic National Space Committee (HNSC)/Greece.
- Hellenic Space Agency (HSA)/Greece.
- Indian Space Research Organization (ISRO)/India.
- Institute of Space Research (IKI)/Russian Federation.
- Korea Aerospace Research Institute (KARI)/Korea.
- Ministry of Communications (MOC)/Israel.
- Mohammed Bin Rashid Space Centre (MBRSC)/United Arab Emirates.
- National Institute of Information and Communications Technology (NICT)/Japan.
- National Oceanic and Atmospheric Administration (NOAA)/USA.
- National Space Agency of the Republic of Kazakhstan (NSARK)/Kazakhstan.
- National Space Organization (NSPO)/Chinese Taipei.
- Naval Center for Space Technology (NCST)/USA.
- Netherlands Space Office (NSO)/The Netherlands.
- Research Institute for Particle & Nuclear Physics (KFKI)/Hungary.
- Scientific and Technological Research Council of Turkey (TUBITAK)/Turkey.
- South African National Space Agency (SANSA)/Republic of South Africa.
- Space and Upper Atmosphere Research Commission (SUPARCO)/Pakistan.
- Swedish Space Corporation (SSC)/Sweden.
- Swiss Space Office (SSO)/Switzerland.
- United States Geological Survey (USGS)/USA.

DOCUMENT CONTROL

Document	Title	Date	Status
CCSDS 120.0-G-1	Lossless Data Compression, Informational Report, Issue 1	May 1997	Original issue, superseded
CCSDS 120.0-G-2	Lossless Data Compression, Informational Report, Issue 2	December 2006	Issue 2, superseded
CCSDS 120.0-G-3	Lossless Data Compression, Informational Report, Issue 3	April 2013	Issue 3, superseded
CCSDS 120.0-G-4	Lossless Data Compression, Informational Report, Issue 4	November 2021	Current issue: – updates material affected by revision of the Lossless Data Compression Blue Book concerning the new file format

CONTENTS

CONTENTS (continued)

Section Page

Figure

CONTENTS (continued)

1 INTRODUCTION

1.1 PURPOSE

This report presents a summary of the key operational concepts and rationale that underlie the requirements for the CCSDS Recommended Standard, *Lossless Data Compression* (reference [1]). Supporting performance information along with illustrations are also included. This report provides a broad tutorial overview of the CCSDS Lossless Data Compression algorithm and is aimed at helping first-time readers to understand the Recommended Standard.

1.2 SCOPE

This document provides supporting and descriptive material only: **it is not part of the Recommended Standard**. In the event of any conflict between the *Lossless Data Compression* Recommended Standard and the material presented herein, the Recommended Standard shall prevail.

1.3 DOCUMENT STRUCTURE

This document is organized as follows.

- Section 2 gives an overview of the compression algorithm formalized in the Recommended Standard. This section is intended to provide high-level information for a user considering using the Recommended Standard, whether the user intends to produce an implementation or obtain one from a third party.
- Section 3 describes the underlying compression algorithm formalized in the Recommended Standard.
- Section 4 addresses several systems issues related to embedding data compression schemes on board a spacecraft.
- Section 5 discusses practical issues relevant to implementation of the standard.
- Section 6 presents compression performance results for the Recommended Standard for eight different scientific imaging and non-imaging instruments.
- Annex A provides links to available software implementations of the Recommended Standard, to test data.
- Annex B provides a list of abbreviations and acronyms used in the text of this document.

1.4 CONVENTION AND DEFINITIONS

The following convention applies throughout this document:

'The Recommended Standard' or 'The Recommendation' refer to the Lossless Data Compression Recommended Standard described in reference [1].

For the purpose of this document, the mathematical notation and definitions described in subsection 1.5 of reference [1] apply.

1.5 REFERENCES

[1] *Lossless Data Compression*. Issue 3. Recommendation for Space Data System Standards (Blue Book), CCSDS 121.0-B-3. Washington, D.C.: CCSDS, August 2020.

[2] *Organization and Processes for the Consultative Committee for Space Data Systems*. Issue 4. CCSDS Record (Yellow Book), CCSDS A02.1-Y-4. Washington, D.C.: CCSDS, April 2014.

[3] *Space Packet Protocol*. Issue 2. Recommendation for Space Data System Standards (Blue Book), CCSDS 133.0-B-2. Washington, D.C.: CCSDS, June 2020.

[4] *AOS Space Data Link Protocol*. Issue 4. Recommendation for Space Data System Standards (Blue Book), CCSDS 732.0-B-4. Washington, D.C.: CCSDS, October 2021.

[5] Pen-Shu Yeh and Warner H. Miller. *Application Guide for Universal Source Coding for Space*. NASA Technical Paper 3441. Washington, DC: NASA, December 1993.

[6] Robert F. Rice, Pen-Shu Yeh, and Warner H. Miller. "Algorithms for High Speed Universal Noiseless Coding." In *Proceedings of AIAA Computing in Aerospace, IX (October 19-21, 1993, San Diego, CA)*. Reston, Virginia: AIAA, 1993.

[7] *TM Synchronization and Channel Coding*. Issue 3. Recommendation for Space Data System Standards (Blue Book), CCSDS 131.0-B-3. Washington, D.C.: CCSDS, September 2017.

[8] Pen-Shu Yeh, Robert Rice, and Warner Miller. *On the Optimality of Code Options for a Universal Noiseless Coder*. NASA/JPL Publication 91-2. Pasadena, California: JPL, February 15, 1991.

[9] Pen-Shu Yeh and Warner H. Miller. "A Real Time Lossless Data Compression Technology for Remote-Sensing and Other Applications." In *Proceeding of IEEE International Symposium on Circuits and Systems, 1995. ISCAS '95 (30 Apr-3 May 1995, Seattle, Washington)*, Vol. 2:1098–1101. New York: IEEE, 1995.

[10] Masud Mansuripur. *Introduction to Information Theory*. Englewood Cliffs, N.J.: Prentice Hall, 1987.

[11] *Low-Complexity Lossless and Near-Lossless Multispectral and Hyperspectral Image Compression*. Issue 2. Recommendation for Space Data System Standards (Blue Book), CCSDS 123.0-B-2. Washington, D.C.: CCSDS, February 2019.

[12] *CCSDS File Delivery Protocol (CFDP)*. Issue 5. Recommendation for Space Data System Standards (Blue Book), CCSDS 727.0-B-5. Washington, D.C.: CCSDS, July 2020.

[13] Nektarios Kranitis, et al. "Efficient Field-Programmable Gate Array Implementation of CCSDS 121.0-B-2 Lossless Data Compression Algorithm for Image Compression." *SPIE Journal of Applied Remote Sensing* 9, no. 1 (19 May 2015).

[14] Yubal Barrios, et al. "SHyLoC 2.0: A Versatile Hardware Solution for On-Board Data and Hyperspectral Image Compression on Future Space Missions." *IEEE Access* 8 (2020): 54269-54287.

[15] R. Vitulli. "PRDC: An ASIC Device for Lossless Data Compression Implementing the Rice Algorithm." In *IGARSS 2004. 2004 IEEE International Geoscience and Remote Sensing Symposium (20–24 September 2004, Anchorage, AK)*, 320. Piscataway, New Jersey: IEEE, 2004.

[16] A. Kiely. "Selecting the Golomb Parameter in Rice Coding." *IPN Progress Report* 42-159 (November 2004).

[17] I. Rodriguez, et al. "On the Embedded GPU Parallelisation of On-Board CCSDS Compressors: A Benchmarking Approach." In *Proceedings of the Seventh International Workshop on On-Board Payload Data Compression (OBPDC 2020) (21-23 September 2020)*. Noordwijk, The Netherlands: ESA, 2020.

[18] *TM Space Data Link Protocol*. Issue 3. Recommendation for Space Data System Standards (Blue Book), CCSDS 132.0-B-3. Washington, D.C.: CCSDS, October 2021.

[19] *Unified Space Data Link Protocol*. Issue 2. Recommendation for Space Data System Standards (Blue Book), CCSDS 732.1-B-2. Washington, D.C.: CCSDS, October 2021.

2 OVERVIEW

Lossless data compression has been suggested for many space science exploration mission applications either to increase the science return or to reduce the requirement for onboard memory, station contact time, and data archival volume. A Lossless compression technique guarantees full reconstruction of the original data without incurring any distortion in the process.

The Lossless Data Compression technique recommended preserves the source data accuracy by removing redundancy from the application source data. In the decompression processes, the original source data is reconstructed from the compressed data by restoring the removed redundancy. The reconstructed data is an exact replica of the original source data. The quantity of redundancy removed from the source data is variable and is highly dependent on the source data statistics, which are often non-stationary.

The Lossless Data Compression algorithm can be applied at the application data source or performed as a function of the onboard data system (see figure 2-1). The performance of the data compression algorithm is independent of where it is applied. However, if the data compression algorithm is part of the onboard data system, the onboard data system will, in general, have to capture the data in a buffer. In both cases, it may be necessary to rearrange the data into appropriate sequence before applying the data compression algorithm. The purpose of rearranging data is to improve the compression ratio.

After compression has been performed, the resulting variable-length data structure can then be packed into data units (packets or files) in order to be transported through a space-to-ground communication link from the source on board a space vehicle to a data sink on the ground data system as shown in figure 2-1. The existing Packet Format defines the packet structure to be used when reference [3] is used for transmission. Issue 3 of the Recommended Standard (reference [1]) also defines an optional File Format intended for data required to be stored or transmitted as a file as indicated in reference [12]. It provides information about compression options and defines a structure to store the sequence of Coded Data Sets (CDSes) resulting from compressing the input samples.

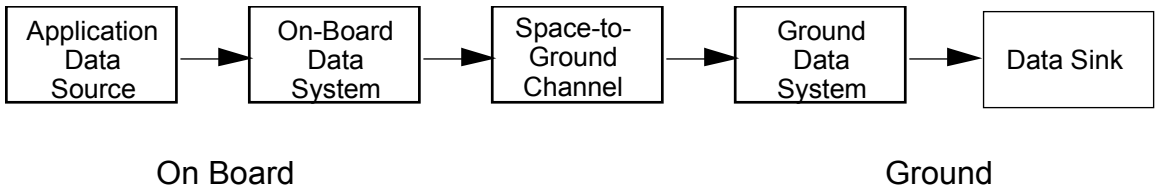


Figure 2-1: Telemetry Data System

The essential purpose of the telemetry system is to permit the application processes on board to create data units in the form of packets or files. These data units are then transmitted over the space-to-ground communication channel in a way that enables the ground system to recover the data with high reliability. The contents of the data units are then extracted and their data are provided to the ground sink in sequence to be decompressed.

This document provides a top-level description of the Rice algorithm architecture, which is the basis for the compression technique (see reference [6]). It then addresses systems issues important for onboard implementation. The latter part of the document provides case-study examples drawn from a broad spectrum of science instruments (see reference [5]). Reference [9] also provides Lossless data compression results obtained from three different algorithms obtained from five data sets.

3 ALGORITHM

3.1 GENERAL

Implementation of a data-compression technique on a spacecraft will benefit from the following considerations:

a) the ability of the algorithm to adapt to changes in data statistics to maximize performance;

b) the requirements to minimize the number of processing steps and reduce memory and power usage;

c) the ability of the algorithm to operate in one pass;

d) the ability of the algorithm to be interfaced with a packet or file data system without requiring a priori information re-establishing data statistics for every packet or file;

e) the ability to limit the effects of a channel error.

There are a few well-known Lossless compression techniques, including Huffman coding, arithmetic coding, Ziv-Lempel coding, and variants of each. After extensive study of the above criteria and performance comparison on a set of test images, the Rice algorithm was selected for the CCSDS Recommended Standard (see reference [9]).

The Rice algorithm uses a set of variable-length codes to achieve compression. Each code is nearly optimal for a particular geometrically distributed source. Variable-length codes, such as Huffman codes and the codes used by the Rice algorithm, compress data by assigning shorter codewords to symbols that are expected to occur with higher frequency, as illustrated in 3.2.1. By using several different codes and transmitting the code identifier, the Rice algorithm can adapt to many sources from low entropy (more compressible) to high entropy (less compressible). Because blocks of source samples are encoded independently, side information does not need to be carried across packet or file boundaries, and the performance of the algorithm is independent of the packet or file size. For a discussion of the term entropy, the reader should consult an information theory textbook such as reference [10].

A block diagram of the Rice algorithm is shown in figure 3-1. It consists of a preprocessor to decorrelate data samples and subsequently map them into symbols suitable for the entropy coding stage. The input data to the preprocessor, $\boldsymbol{x}$, is a J-sample block of n-bit samples:

$$\boldsymbol{x} = x_1, x_2, \ldots x_J.$$

The preprocessor transforms input data into blocks of preprocessed samples, $\boldsymbol{\delta}$, where:

$$\boldsymbol{\delta} = \delta_1, \delta_2, \ldots \delta_J.$$

The Adaptive Encoder converts preprocessed samples, $\boldsymbol{\delta}$, into an encoded bit sequence $\boldsymbol{y}$.

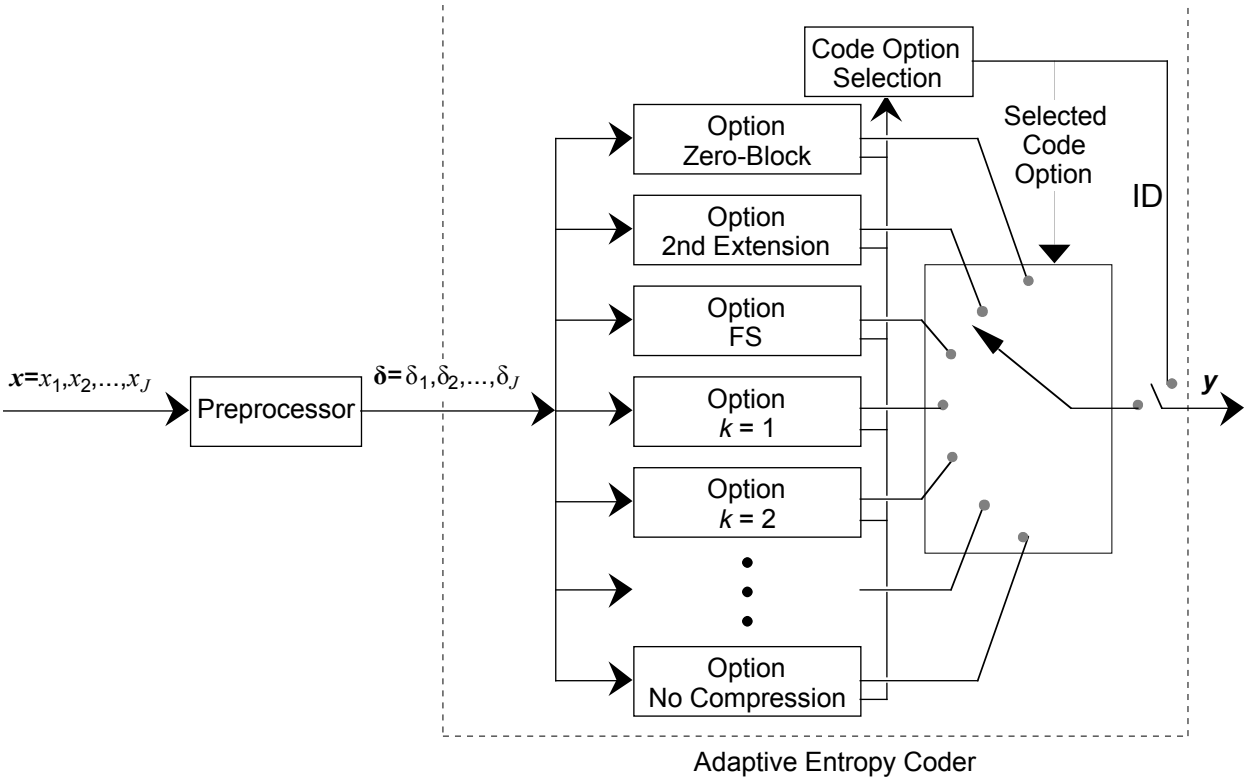


Figure 3-1: The Encoder Architecture

The entropy coding module consists of a collection of variable-length codes that can be applied to each block of J preprocessed samples. For each block, the coding option achieving the best compression is selected to encode the block. The encoded block is preceded by an ID bit pattern that identifies the coding option to the decoder. Because a new code option can be selected for each block, the Rice algorithm can adapt to changing source statistics. Thus shorter values of the block length parameter J allow faster adaptation to changing source statistics. However, the fraction of encoded bits used to indicate the coding option (the 'overhead' associated with code option identification) decreases for larger values of J.

The first issue of the Recommended Standard allowed J to take values {8, 16}. In practice, longer block lengths often offer improved overall compression effectiveness (because of reduced overhead). Motivated by this fact, especially when the Recommended Standard is used as the entropy coding option for multispectral or hyperspectral image compression as specified in reference [11], issues 2 and 3 of the Recommended Standard expand the allowed values of J to {8, 16, 32, 64}.

3.2 ENTROPY CODER

3.2.1 FUNDAMENTAL SEQUENCE ENCODING

To see how a variable length code can achieve data compression, consider two methods of encoding sequences from an alphabet of four symbols:

Symbol	Code 1	Code 2 (FS Code)
s_1	00	1
s_2	01	01
s_3	10	001
s_4	11	0001

It should be noted that the Code 2 is the Fundamental Sequence (FS) code.

Under either code, encoded bit streams corresponding to symbol sequences are uniquely decodable. For fixed-length Code 1, it is recognized that every two encoded bits correspond to a symbol. For the variable-length FS code, it is recognized that each '1' digit signals the end of a codeword, and the number of preceding zeros identifies which symbol was transmitted. This simple decoding procedure allows FS codewords to be decoded without the use of lookup tables.

The reason that the FS code can achieve compression is that when symbol s_1 occurs very frequently, and symbols s_3 and s_4 are very rare, on average fewer than two encoded bits per symbol will be will transmitted, whereas Code 1 always requires two encoded bits per symbol. Longer FS codes achieve compression in a similar manner.

3.2.2 THE SPLIT-SAMPLE OPTIONS

Most of the options in the entropy coder are called 'split-sample options'. The k^{th} split-sample option takes a block of J preprocessed data samples, splits off the k least significant bits from each sample and encodes the remaining higher order bits with a simple FS codeword before appending the split bits to the encoded FS data stream. Each split-sample option in the Rice algorithm is designed to produce compressed data with a range of about 1 bit/sample (approximately $k + 1.5$ to $k + 2.5$ bits/sample); the code option yielding the fewest encoded bits will be chosen for each block by the option-select logic. This option selection process assures that the block will be coded with the best available code option on the same block of data, but this does not necessarily imply that the source entropy lies in that range. The actual source entropy value could be lower; the source statistics and the effectiveness of the preprocessing stage determine how closely entropy can be approached.

A theoretical expected codeword length of the split-sample coding scheme is shown in figure 3-2 for various values of k, where $k = 0$ is the fundamental sequence option. These

curves are obtained under the assumption that the source approaches a discrete geometric distribution. With this source model, tracing only the portions of these curves that are closest to the ideal diagonal performance curve at any given entropy will result in a theoretically projected performance curve shown in figure 3-3. For a practical system shown in figure 3-1, ID bits of up to 5-bit length will be needed for every block of J samples as defined in table 5-1 of reference [1]. In such a system, the source is likely to deviate from the discrete geometric distribution; however, the option with the fewest coded bits will be selected. Actual coding results on aerial imagery, along with the coder's applicability to other data sources with a Gaussian or a Poisson probability distribution function, are provided in reference [8].

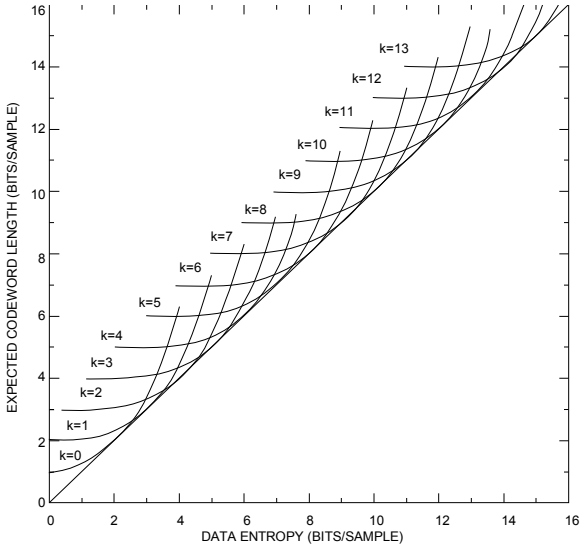


Figure 3-2: Performance Curve for Various *k*

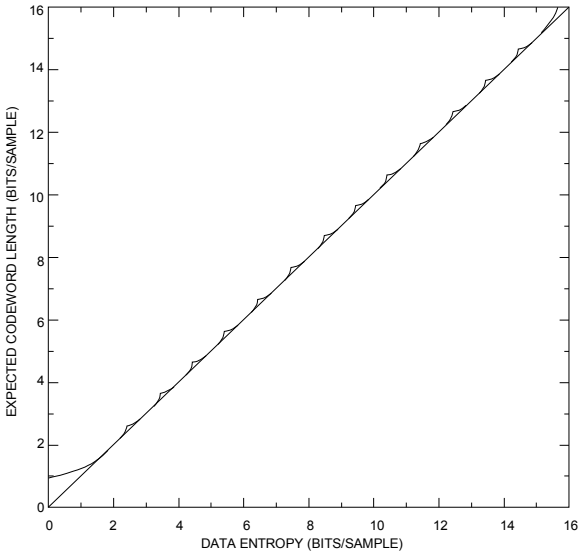


Figure 3-3: Effective Performance Curve

3.2.3 LOW ENTROPY OPTIONS

3.2.3.1 Overview

To provide compressed data rates below 1.5 bit/sample, the Recommended Standard includes two low-entropy options. The low-entropy options are particularly efficient when the preprocessed samples consist only of small values. These low-entropy options are the Second Extension option and Zero Block option.

3.2.3.2 Second Extension Option

The Second-Extension option is designed to produce compressed data rates in the range of 0.5 bits/sample to 1.5 bits/sample. When this option is selected, the encoding scheme first pairs consecutive preprocessed samples of the *J*-sample block and then transforms the paired samples into a new value that is coded with an FS codeword. The FS codeword for γ is transmitted, where:

$$\gamma_j = (\delta_{2j-1} + \delta_{2j})\,(\delta_{2j-1} + \delta_{2j}+1)/2 + \delta_{2j},$$

where $j = 1, 2, \ldots J/2$.

3.2.3.3 Zero-Block Option

The Zero-Block option is selected when one or more blocks of preprocessed samples are all zeros. In this case, the number of adjacent all-zero preprocessed blocks are encoded by an FS codeword as shown in table 3-1.

Table 3-1: Zero-Block Fundamental Sequence Codewords as a Function of the Number of Consecutive All-Zeros Blocks

Number of All-Zeros Blocks	FS Codeword
1	1
2	01
3	001
4	0001
ROS	00001
5	000001
6	0000001
7	00000001
8	000000001
.	.
.	.
.	.
63	0000 . . . 0000000001 (63 0s and a 1)

The *r* blocks between consecutive reference samples are divided into segments, as described in subsection 3.5.2 of the *Lossless Data Compression* Recommended Standard (reference [1]). Each segment, except possibly the last, contains 64 blocks. The Remainder-of-Segment (ROS) codeword is used to denote that the remainder of a segment consists of five or more all-zeros blocks. Even when reference samples are not used, the parameter *r* is still used to segment the data for the Zero-Block option.

For example, if there are 200 blocks between consecutive reference samples, the first three segments have 64 blocks and the fourth segment has 8 blocks. During calibration of sensors, all 200 blocks might have the same values. After a unit-delay predictor is applied, the corresponding 200 blocks of preprocessed samples will have all-zero values except for the first sample in the first block, which is the reference sample. The relation between the uncoded segments and the coded segments is depicted in figure 3-4. The ROS condition is signaled at the end of each 64-block segment, and a new ID is issued at the beginning of each segment. It should be noted that the ROS condition also encodes the last eight all-zero blocks.

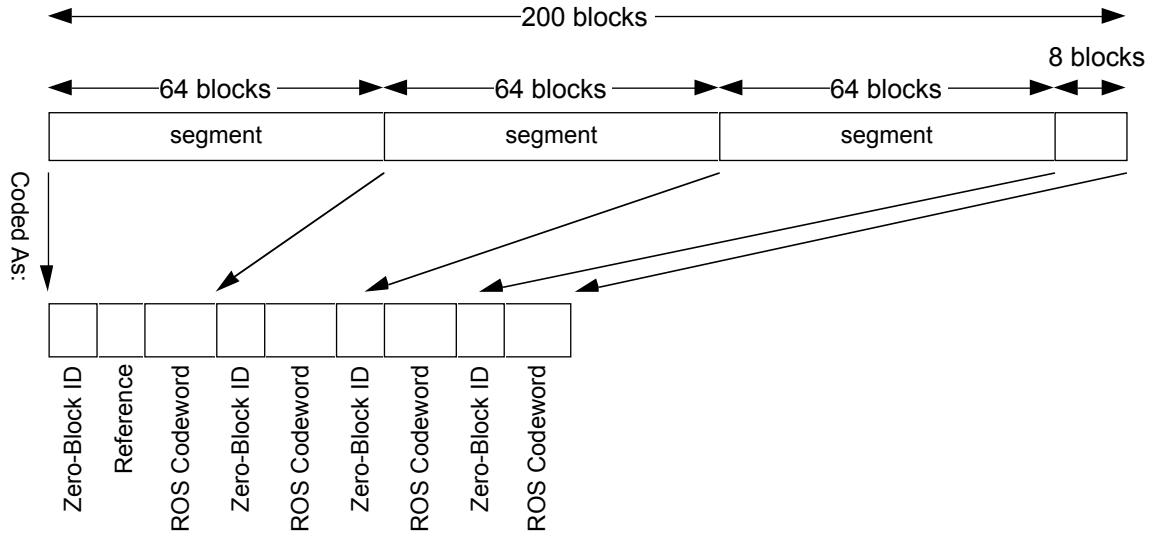


Figure 3-4: Example of All-Zero Blocks

3.2.4 NO-COMPRESSION OPTION

When none of the above options provides any data compression on a block, the no-compression option is selected. Under this option, the preprocessed block of data is transmitted without alteration except for a prefixed identifier.

3.2.5 OPTION SELECTION

For a block of samples, the entropy coder selects the option that minimizes the number of encoded bits. This includes the identifier bit sequence that specifies which code option was selected.

3.2.6 OPTION IDENTIFICATION

When the source dynamic range is small (i.e., for small values of resolution n), code options corresponding to larger values of k are not useful. For this reason, when $n \leq 4$, a user can choose to use the Restricted set of coding options (see table 5-1 of reference [1]), which limits the available coding options and thus reduces the number of bits used to identify the coding option. This reduces overhead and improves compression performance for certain low dynamic range data sources.

3.3 THE PREPROCESSOR

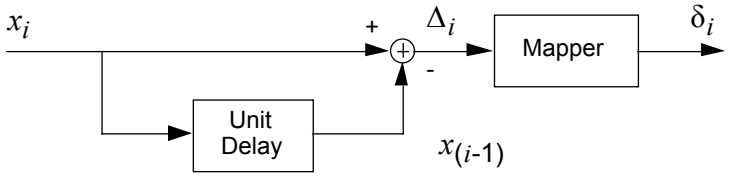
3.3.1 PREDICTOR

The role of the preprocessor is to transform the data into samples that can be more efficiently compressed by the entropy encoder. To ensure that compression is Lossless, the preprocessor must be reversible. It can be seen from the example in 3.2.1 that to achieve effective compression there needs to be a preprocessing stage that transforms the original data so that shorter codewords occur with higher probability than longer codewords. If there are several candidate preprocessing stages, it would be preferable to select the one that produces the shortest average codeword length.

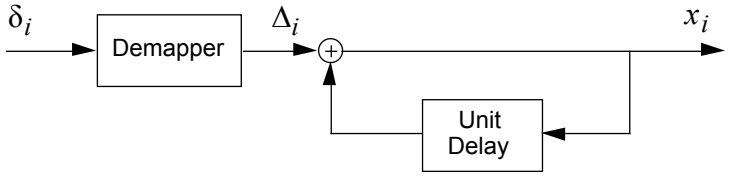
In general, a preprocessor that removes correlation between samples in the input data block will improve the performance of the entropy coder. In the following discussion, it is assumed that preprocessing is done by a predictor followed by a prediction error mapper. For some types of data, more sophisticated transform-based techniques can offer improved compression efficiency, at the expense of higher complexity.

The decorrelation function of the preprocessor can be implemented by a judicious choice of the predictor for an expected set of data samples. The selection of a predictor should take into account the expected data as well as possible variations in the background noise and the gain of the sensors acquiring the data. The predictor should be chosen to minimize the amount of noise resulting from sensor non-uniformity.

One of the simplest predictive coders is a linear first-order unit-delay predictor shown in figure 3-5a. The output, Δ_i, will be the difference between the input sample and the predicted value which, when using the unit-delay predictor, is equal to the preceding sample value except for the first sample in a reference interval (as defined in 4.3 of reference [1]) for which the predicted value is the current sample value, *xi*.



a) Preprocessor



b) Postprocessor

Figure 3-5: Block Diagram of Unit-Delay Predictive Preprocessor and Postprocessor

Several other predictors exist. In general, users may select any prediction scheme that best decorrelates the input data stream. For example, consider an image with intensity values $x(i, j)$, where i represents the scan line and j the pixel number within the scan (see figure 3-6). Possible predictor types include:

a) One-dimensional first-order predictor:

   The predicted intensity value, $\hat{x}(i, j)$, might be equal to the previous samples on the same scan line, $x(i, j–1)$, or the neighboring sample value from a previous scan line, $x(i–1, j)$. The unit-delay predictor is an example of a one-dimensional first-order predictor.

b) Two-dimensional second-order predictor:

   The predicted intensity value, $\hat{x}(i, j)$, might be the average of the adjacent samples, $x(i, j–1)$ and $x(i–1, j)$.

c) Two-dimensional third-order predictor:

   The predicted intensity value, $\hat{x}(i, j)$, might be equal to a weighted combination of three neighboring values of $x(i, j–1)$, $x(i–1, j)$, and $x(i–1, j–1)$.

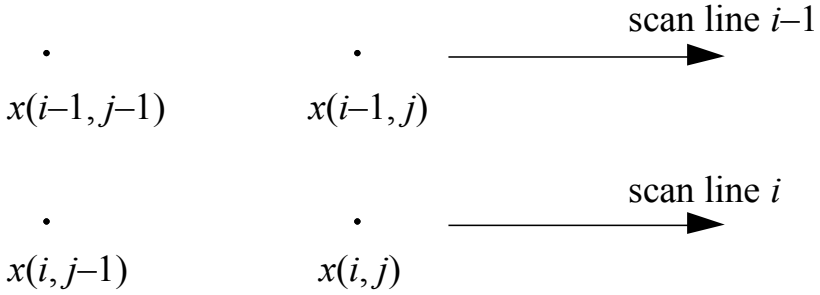


Figure 3-6: Two-Dimensional Predictor of an Image Where $x(i, j)$ Is the Sample to Be Predicted

For Multispectral data, another prediction technique is to use samples from one spectral band as an input to a higher order prediction function for the next band. Nevertheless, users handling Multispectral or Hyperspectral data should consider using the more specialized Recommended Standard defined in reference [11] to obtain better performance.

3.3.2 REFERENCE SAMPLE

A reference sample is an unaltered data sample upon which successive predictions are based. Reference samples are required by the decoder to recover the original values from difference values. When, and only when, a Unit-Delay Predictor or other higher-order predictor that bases its predictions on previous sample values is used, reference samples are required by the decoder in order to invert the preprocessing function. Otherwise, reference samples are not employed. The user must determine how often to insert references. When required, the

reference must be the first sample of a block of J input data samples. In packetized formats, the reference sample must be in the first CDS in the packet data field, as defined in 4.2.6 of reference [1], and must repeat after every r blocks of data samples.

3.3.3 THE MAPPER

Based on the predicted value, $\hat{x}_i$, the prediction error mapper converts each prediction error value, Δ_i, to an n-bit nonnegative integer, δ_i, suitable for processing by the entropy coder. For most efficient compression by the entropy coding stage, the preprocessed symbols, δ_i, should satisfy:

$$p_0 \geq p_1 \geq p_2 \geq \cdots p_j \geq \cdots p_{(2^n-1)},$$

where p_j is the probability that δ_i equals integer j. This ensures that more probable symbols are encoded with shorter codewords.

The following example illustrates the operation of the prediction error mapper after a unit-delay predictor is applied to 8-bit data samples of values from 0 to 255:

Sample Value x_i	Predictor Value $\hat{x}_i$	Δ_i	θ_i	δ_i
101	—	—	—	—
101	101	0	101	0
100	101	–1	101	1
101	100	1	100	2
99	101	–2	101	3
101	99	2	99	4
223	101	122	101	223
100	223	–123	32	155

If x_{min} and x_{max} denote the minimum and maximum values of any input sample x_i, then clearly any reasonable predicted value $\hat{x}_i$ must lie in the range $[x_{min}, x_{max}]$. Consequently, the prediction error value Δ_i must be one of the 2^n values in the range $[x_{min} - \hat{x}_i, x_{max} - \hat{x}_i]$, as illustrated in figure 3-7a. It is expected that for a well-chosen predictor, small values of $|\Delta_i|$ are more likely than large values, as shown in figure 3-7b. Consequently, the prediction error mapping function:

$$\delta_i = \begin{cases} 2\Delta_i & 0 \leq \Delta_i \leq \theta_i \\ 2|\Delta_i| - 1 & -\theta_i \leq \Delta_i \leq 0, \\ \theta_i + |\Delta_i| & \text{otherwise} \end{cases}$$

$$\text{where } \theta_i = \min(\hat{x}_i - x_{min}, x_{max} - \hat{x}_i),$$

has the property that $p_i < p_j$ whenever $|\Delta_i| > |\Delta_j|$. This property increases the likelihood that the probability ordering of p_i (described above) is satisfied.

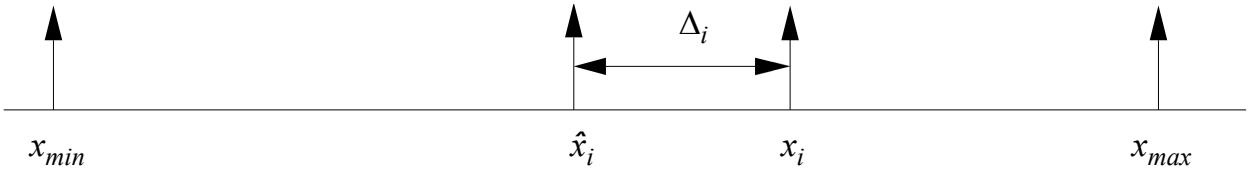


Figure 3-7a: Relative Position of Sample Value x_i and Predictor Value $\hat{x}_i$

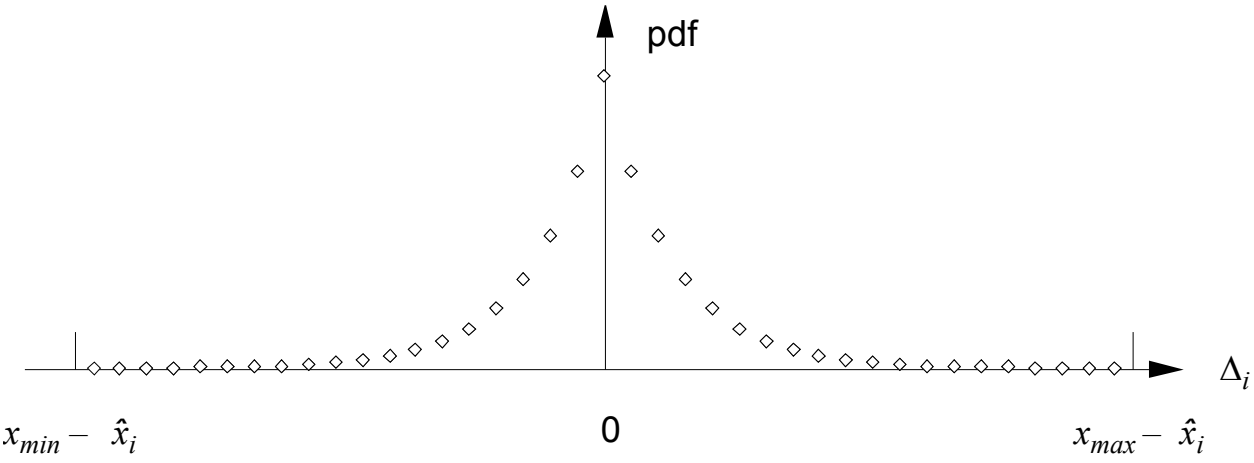


Figure 3-7b: Typical Probability Distribution Function of Δ_i for Imaging Data

4 SYSTEMS ISSUES

4.1 INTRODUCTION

Several systems issues related to embedding data compression schemes on board a spacecraft should be addressed. These include: the relation between the focal-plane array arrangement, data sampling approach, and prediction direction; the number of samples per packet or file; the subsequent data packetizing scheme and how it relates to error propagation when a bit error occurs in the communication channel; and the amount of smoothing buffer required when the compressed data are passed to a constant-rate communication link.

4.2 SENSOR CHARACTERISTICS

Advanced imaging instruments and spectrometers often use arrays of detectors arranged in a one-dimensional or two-dimensional configuration; one example is the Charge-Coupled Device (CCD). This array of individual detectors tends to have slight differences in response to the same input photon intensity. For instance, CCDs usually have a different gain and dark current value for each detector. It is important to have the sensors calibrated so that the data reflects the actual signals received by the sensor. Investigation of 8-bit resolution CCD sensor data with a gain variation of 7 percent and dark current level of less than one half-bit reveals an entropy variation of 2.8 bits, which renders the prediction scheme ineffective for data compression (i.e., a compression ratio of 2.6 versus 1.4). To minimize the effect of detector gain variation on the compression gain in a one-dimensional prediction scheme, the preprocessor (figure 3-5) will be much more effective when data acquired on one detector element are used as the predictor for data acquired on the same detector whose characteristics are stationary. An example is the push-broom scheme, used in many Earth-viewing satellite systems, depicted in figure 4-1. An array of detectors is arranged in the cross-track direction, which is also the detector data read-out direction. The scanning of the ground scene is achieved by the motion of the satellite in the along-track direction. In such configuration, predictive coding will be more effective in the along-track direction, in which data from the same detector is used as the predictor.

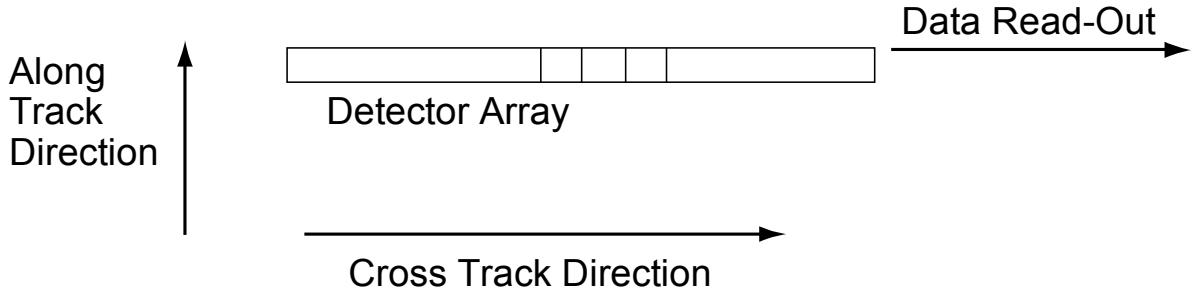


Figure 4-1: Push-Broom Scanning Scheme

4.3 ERROR PROPAGATION

A major concern in using data compression is the possibility of error propagation in the event of a bit error, which can lead to a reconstruction error over successive data points. This effect can be reduced by packetization or by using files, to limit the extent of such error propagation, and channel coding such as Reed-Solomon coding, to reduce the probability of a channel error (see reference [7]). An example for space application would be to packetize the compressed data of one scan line in a CCSDS-recommended data packet format (reference [3]) and the compressed data of the next scan line into a second packet. In the case of a bit error in one packet, the decompression error would only propagate from the location of the error event to the end of the packet.

4.4 DATA OUTPUT STRUCTURES AND COMPRESSION PERFORMANCE

The total number of encoded bits resulting from Losslessly compressing a fixed number of data samples is variable. The CCSDS data architecture provides methods to transmit the (variable-length size) encoded bits forming a data stream, which can be then inserted into a fixed-length transfer frame to be transported from space to ground. Use of packets or files is essential for preventing error propagation as mentioned above, but it also affects compression performance, as increased robustness to data loss/errors generally requires increased overhead.

For better compression performance, the data units (packets or files) should comprise as many blocks as possible to minimize the impact of the header overhead. Because of the insertion of configuration information, the use of the optional CIP or File Format, defined in reference [1], results in extra overhead with respect to the plain insertion into space packets.

There exist other available compression algorithms whose performance depends on establishing the long-term statistics of the collection of sensor data. Such schemes will in general give good compression performance for a large amount of data, and poor performance for a relatively small amount of data. And if packets or files are used as a means of preventing error propagation in conjunction with these algorithms, one would expect poor performance for a shorter data unit and a better performance for a larger data unit. The drawback is that the loss of data due to error propagation may be intolerable for the larger data units. Nevertheless, it must be noted that by using the CFDP protocol in reference [12], loss of data can be prevented by retransmission of non-received data.

Lossless data compression is not recommended for systems with fixed-length data structures because the compression performance achieved is lower than that achieved with a variable-length packet system. For onboard designs that require fixed-length packets, the compressed data will be followed by fill bits to bring the packet to its fixed length. The quantity of fill bits added should insure that packet overflow is highly improbable. If the compressed bit count is greater than the fixed length, truncation occurs with loss of data. The decoder recognizes the truncation condition and signals when a block has been truncated by outputting dummy symbols. The decoder will output as many dummy symbols as would be in a typical packet that was not truncated. If one or more blocks were truncated,

corresponding blocks of dummy symbols will be output to fill out the required blocks-per-packet. To lessen the effects of truncation when fixed-length packet structures are used, the design could be engineered so that the split *k*-bit Least Significant Bits (LSBs) are the last to be transmitted. In such an arrangement, if truncation occurs, probably only the LSBs would be lost, and thus the impact on performance would be lessened. However, for the packet, the compression is no longer Lossless.

4.5 BUFFER REQUIREMENT

Depending on the mission requirement, the variable-length data units resulting from packing the compressed bit stream into files or CCSDS packets can be stored in a large onboard memory, or they can be multiplexed with other data units before being stored in the memory device. In either case, the large-capacity memory device lessens the variation in the data-unit length. Subsequent readout from the memory is performed at a fixed rate. In other situations, the variable-length data units may have to be temporarily buffered before a direct transmission to the communication link. The temporary buffer serves as a smoothing buffer for the link. Occasionally, fill bits are inserted in the data stream to provide a constant readout rate. The amount of buffer needed is a function of the incoming rate, the data-unit-length statistics, and the readout rate.

An analysis was conducted, using the CCSDS *AOS Space Data Link Protocol* (reference [4]), which examined issues such as the maximum smoothing buffer size requirement when a variable-rate Multiplexing Protocol Data Unit (MPDU) is transmitted to ground within a fixed-rate Virtual Channel Data Unit (VCDU). In this analysis, the MPDU transported variable-length packets with compressed data whose length varied with Gaussian statistics. Figure 4-2 shows the results of this analysis, where the probability of buffer overflow decreases as the buffer size increases. Doubling the buffer size normally will decrease the probability of buffer overflow by several orders of magnitude. The MPDUs are shifted asynchronously into the VCDU whenever they are filled, and thus the MPDU input to the VCDU occurs at a packet generation time period of t_p, when it is available; t_s represents the sampling period of the VCDU, which provides a constant data rate to the physical channel.

NOTE – Even though this analysis was conducted only with *AOS Space Data Link Protocol* (reference [4]), the Recommended Standard can also be considered for data systems using the *TM Space Data Link Protocol* (reference [18]) and the *Unified Space Data Link Protocol* (reference [19]).

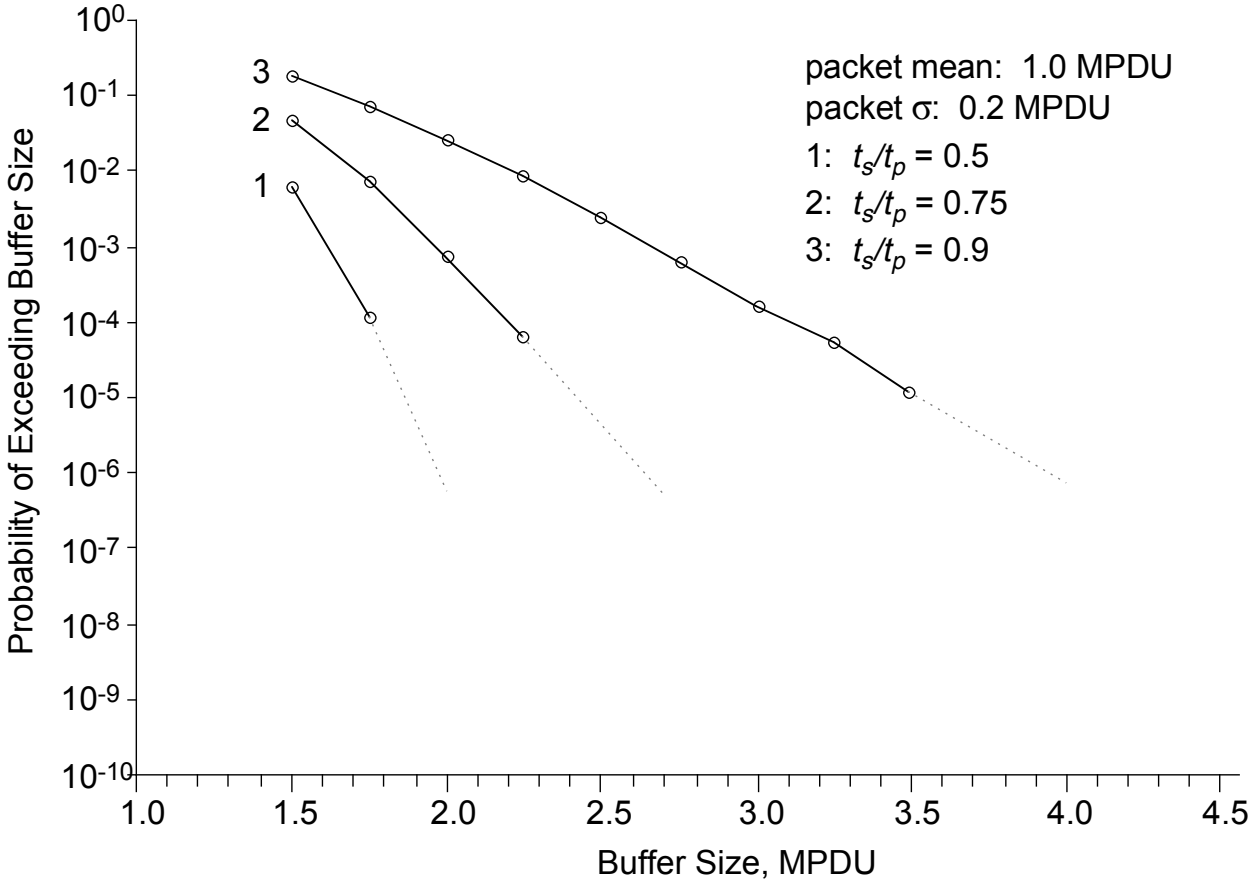


Figure 4-2: Probability of Buffer Overflow

4.6 FILE FORMAT

In deploying file-based operations for space missions, there has been an increasing need among the space agencies to simplify the access to the space segment and ultimately the overall operations. The use of a robust protocol suitable for transmission of files to and from data storage mediums over a ground-space communication link is therefore key for the adoption of file-based operations. Accordingly, several missions under preparation have selected for implementation the CCSDS File Delivery Protocol (CFDP) (reference [12]), which has motivated the definition of a File Format in issue 3 of the Recommended Standard (reference [1]).

Nevertheless, the previous versions of this Recommended Standard are also compatible with the use of the CCSDS File Delivery Protocol, as the file management can be implemented by a different entity. The new File Format simply provides a more straightforward way to benefit from the file services provided by the CFDP protocol or other alternative file-based systems.

The File Format includes a File Header that contains the necessary information for a decoder to decompress data, including compression parameters and description of the representation of the input samples.

5 IMPLEMENTATION

5.1 OVERVIEW

The examples shown in this section are based on a system using the CCSDS *AOS Space Data Link Protocol* (reference [4]) in conjunction with the CCSDS *Space Packet Protocol* (reference [3]) and the CCSDS *TM Synchronization and Channel Coding* (reference [7]). These examples are generally applicable to the File Format when considering the same protocols as underlying delivery services (compressed files become file data PDUs as described in reference [12] and are then encapsulated over Service Data Units (SDUs) in the Network Layer as defined below).

5.2 SPACE SEGMENT

Figure 5-1 represents the space segment as a functional block diagram that identifies each of the operations related to the data as they are transported through the CCSDS onboard layered architecture when Reed-Solomon error-correction is used. The following description assumes that the onboard instrument data to be compressed is a multispectral imager.

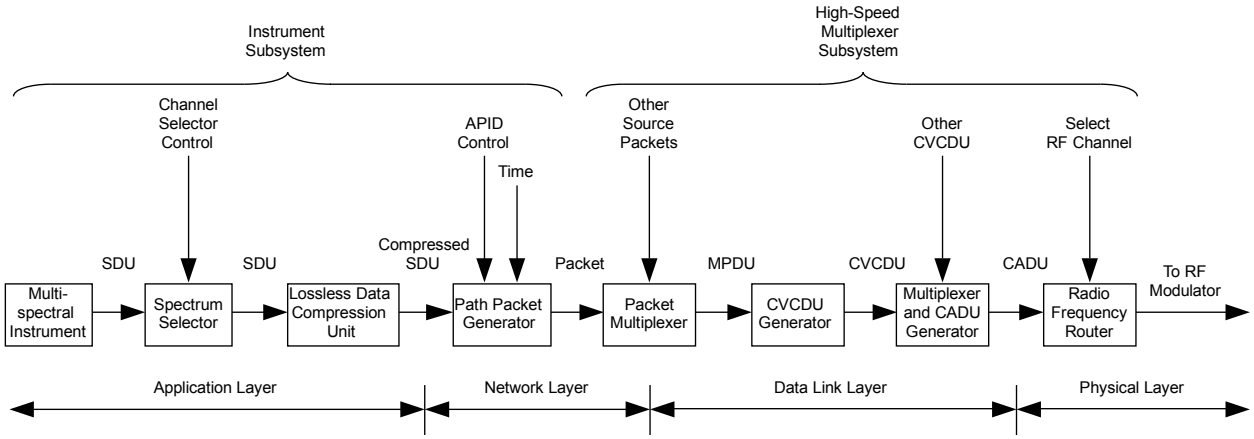


Figure 5-1: The Space Segment

In the Application Layer, spectral channels are selected for the images to be transported for intended users. The selector outputs SDUs, which are metered out in scan lines as they are generated. There will be one SDU for each individual spectral channel (i.e., the scan lines for spectral channels are not multiplexed into a single SDU). Other functions of the Application Layer are to perform the data compression operation on the SDU and to identify the length of the SDU to the nearest octet boundary before passing it to the path packet generator.

In the Network Layer, SDUs are tagged with appropriate Application Process Identifiers (APIDs), and packets are generated with the packet length signaled in the packet header. If time tagging is required, the secondary header flag is set, and time is inserted into the secondary header field. In the Data Link Layer, the image data packets are multiplexed with packets from other instruments into a fixed-length MPDU. The MPDU header contains the

first packet header pointer, which is required for demultiplexing. These MPDUs are then assigned to a particular virtual channel in which Coded Virtual Channel Data Units (CVCDUs) of fixed-length units of octets are generated. The CVCDUs contain the sequence counter and the Reed-Solomon parity octets. These CVCDUs are then randomized to insure bit transition and are passed on to the Multiplexer and Channel Assess Data Unit (CADU) generator. At this point, CADUs from other non-related instrument data are multiplexed together and are separated into selected channels to be routed to users.

5.3 GROUND SEGMENT

Figure 5-2 shows a functional block diagram of the user ground segment.

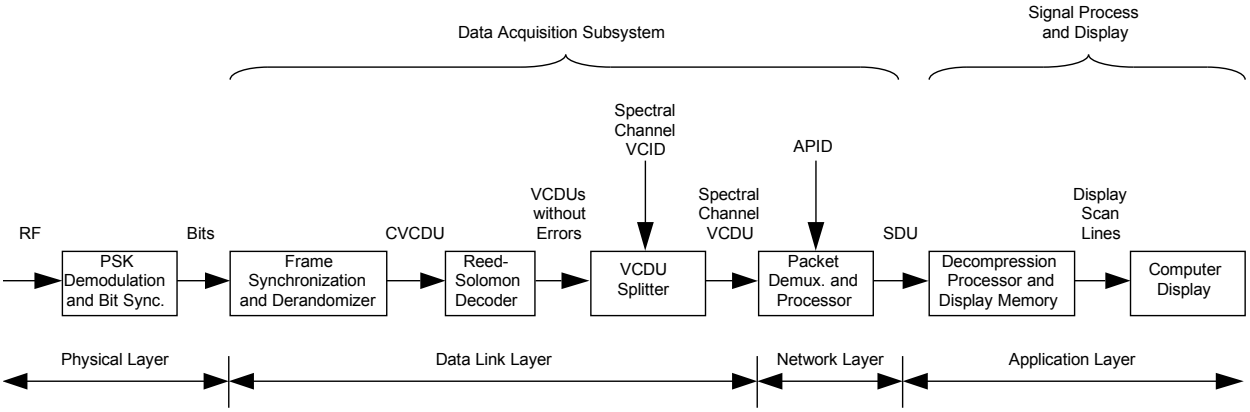


Figure 5-2: Ground Segment

The Physical Layer detects the Radio Frequency (RF) signal; Phase Shift Key (PSK) demodulates and bit synchronizes to the noisy signal. The detected Non-Return to Zero (NRZ) bits and a synchronous clock are passed to the Data Acquisition Subsystem.

The Data Link Layer first accepts the bit string, and then using the knowledge of frame synchronization pattern and frame length, it detects the CADUs and derandomizes the CVCDUs. The CVCDUs are further deinterleaved and Reed-Solomon decoded to correct channel errors. The Reed-Solomon decoder outputs 'error free' VCDUs, which are then passed on to the VCDU splitter. The VCDU splitter receives a string of VCDUs and sorts out the spectral image VCDUs. It also checks the sequence count and passes this information on to the packet processor.

The packet demultiplexer and processor identifies the location of the first packet header in the MPDU and proceeds to process the packet. Administrative and telemetry packets are first decoded to determine the onboard Instrument Subsystem configuration and status. These administrative messages will enable the Ground Acquisition Subsystem to configure itself to properly sort the incoming data and to know what spectral channels are present in the data. From the packet header are determined the packet length, the APID, and the status of the secondary header. If the APID identifies the packet as a compressed spectral image unit, the SDU is extracted and passed on to the decompression processor. If there is a secondary

header field, it is also extracted and used to determine the image generation time. The Decompression Processor and Display Memory accepts the compressed spectral image SDUs, decompresses them in real time, and then inserts the decompressed image lines into the appropriate display-memory buffer.

5.4 DECODING

The Lossless decoder consists of two separate functional parts, the post processor and the adaptive entropy decoder, as shown in figure 5-3. The postprocessor performs both the inverse prediction operation and the inverse of the standard mapper operation (see figure 3-5).

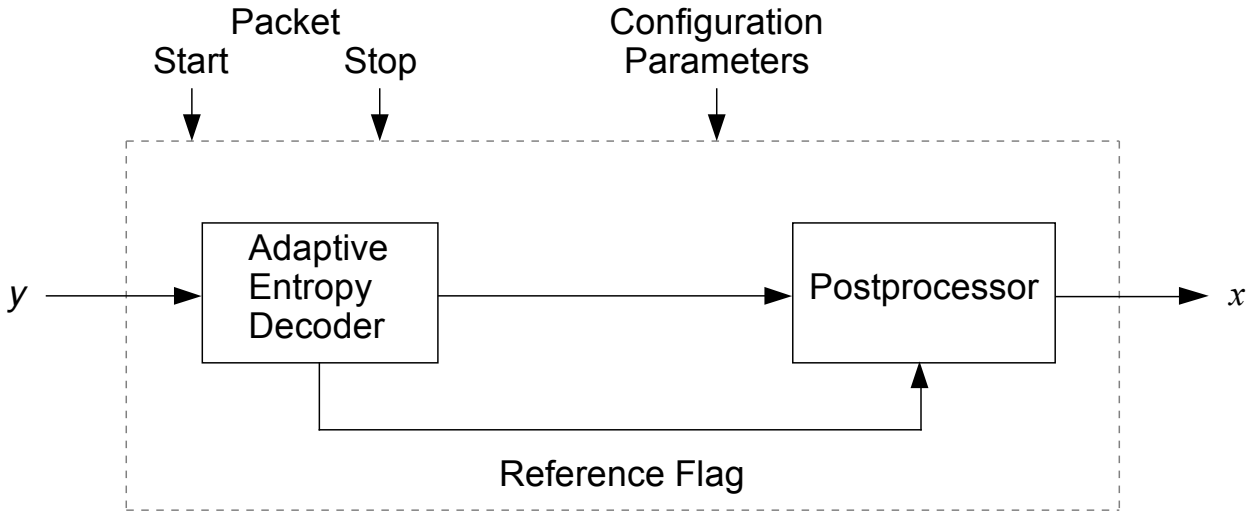


Figure 5-3: Decoder Block Diagram

The first step toward decoding is to set up the configuration parameters so both encoder and decoder operate in the same mode. These configuration parameters will be mission specific and fixed for a given APID, and are either known a priori or are signaled in the Compression Identification Packet (CIP) or in the File Header.

The configuration parameters common to both encoder and decoder are:

- resolution in bits per sample (n = 1,..., 32);
- block size in samples (J = 8, 16, 32, or 64);
- Output Word Size when using the File Format;
- number of CDSes contained in a packet (l = 1,..., 4096) or number of input samples N contained in a file;
- reference sample interval (r = 1,..., 4096);
- preprocessor status (absent/present);

– predictor type if present;

– mapper type;

– data sense (positive or two's complement).

The inputs to the decoder for packets are:

a) the source packet data field containing the compressed source data *y*;

b) the source packet data field start and stop signals; and

c) the decoder configuration parameters.

When using the File Format, all the inputs required by the decoder are included within the File Header (configuration parameters) and the File Body (compressed source data).

The Adaptive Entropy Decoder utilizes the decoder configuration parameters to process the compressed variable-length data contained within the source packet data field or the File Body.

The selected code-option ID bits, which are at the beginning of the CDS, will be extracted first. If a reference sample was inserted in the CDS during encoding, it will always appear in the first CDS of a packet of file and will re-appear after *r* CDSes. The reference word will be extracted next from the CDS bit stream and passed on to the postprocessor. Depending on the ID bits, the decoder will interpret the remaining bits in the CDS as coded bits by one of the options specified in reference [1].

For the No-Compression option, the remaining bits in the CDS will be passed to the postprocessor directly. For all other options, the FS codewords will be decoded as described in 3.2.1. For Split Sample options, each of the extracted *J* (or *J*–1 for a CDS with reference sample) FS codewords will be concatenated with *k* split bits extracted from the stream using the CDS data format in 5.2.3 of reference [1].

For the Second-Extension option, only *J*/2 FS codewords will be extracted. These are then decoded using the following steps:

Step 1: obtain γ_j by counting the number of 0s in the FS codeword;
Step 2: obtain β and m_s using table 5-1 logic;
Step 3: calculate $\delta_{2j} = \gamma_j - m_s$;
Step 4: calculate $\delta_{2j-1} = \beta - \delta_{2j}$.

Table 5-1: Decoding Logic for the Second-Extension Option

γ	β	m_s
0	0	0
1, 2	1	1
3, 4, 5	2	3
6, …, 9	3	6
10, …, 14	4	10
15, …, 20	5	15
21, …, 27	6	21
28, …, 35	7	28
36, …, 44	8	36

The Second-Extension option is expected to be the selected encoding option only for low entropy data. In that case, the number of entries needed in table 5-1 depends on the value of J. For $J = 16$, $\beta \leq 4$ and for $J = 64$, $\beta \leq 8$. However, in case no other encoding options are implemented, this option might be used for higher entropy data. In that case, table 5-1 can be extended for every new i row as follows:

$\gamma_{i,0}$, $\gamma_{i,1}$, … , γ_{i,β_i}	$\beta_i = \beta_{i-1}+1$	$m_{s\,i} = m_{s\,i-1}+ \beta_i$

where $\gamma_{i,0} = \gamma_{i-1,\beta_{i-1}} + 1$ and $\gamma_{i,\beta_i} = \gamma_{i-1,\beta_{i-1}} + \beta_i + 1$;

For the Zero-Block block option, the single FS codeword following the ID and reference indicates the number of all-zero blocks or the ROS condition listed in table 3-1. Using this table, an appropriate number of all-zero blocks are generated as input to the postprocessor.

The postprocessor reverses the mapper function, given the predicted value $\hat{x}_i$. When the preprocessor is not used during encoding, the postprocessor is bypassed as well. The inverse mapper function can be expressed as:

if $\delta_i \leq 2\theta_i$,

$$\Delta_i = \begin{cases} \delta_i/2 & \text{when } \delta_i \text{ is even} \\ -(\delta_i + 1)/2 & \text{when } \delta_i \text{ is odd} \end{cases}$$

if $\delta_i > 2\theta_i$,

$$\Delta_i = \begin{cases} \delta_i - \theta_i & \text{when } \theta_i = \hat{x}_i - x_{min} \\ \theta_i - \delta_i & \text{when } \theta_i = x_{max} - \hat{x}_i \end{cases},$$

where $\theta_i = \min(\hat{x}_i - x_{min}, x_{max} - \hat{x}_i)$

There will be J prediction error values $\Delta_1, \Delta_2, \ldots \Delta_J$ generated from each data block. These values will be used in the unit-delay postprocessor (figure 3-5) to recover the J sample values $x_1, x_2, \ldots x_J$.

If the packet format is used, to decode packets that may include fill bits, two pieces of information must be communicated to the decoder. First, the decoder must know the block size, J, and the number of CDSes in a packet, l. Second, the decoder must know when the Source Packet Data Field (SPDF) begins and when it ends. From these two pieces of information, the number of fill bits at the end of the data field can be determined.

The number of valid samples in the last CDS may be less than J. If so, the user will have added fill samples to the last CDS to make a full J-sample block, and the user must extract the valid samples from the last J-sample block.

If File Format is used, the number of valid samples in the last J-sample block and the fill bits can be easily obtained from the information provided in the header, in particular, the number of input samples, N, and Output Word Size, B, fields. Once the decompressor has obtained N decompressed samples, the additional decompressed samples needed to fill a J-sample block are fill samples and must be discarded. From there, any additional bit up to an Output Word Size boundary is a fill bit. This enables the concatenation of files forming a compressed stream, without the need to indicate where one finishes and the next one starts.

5.5 HARDWARE IMPLEMENTATION

Several hardware implementations of the Recommended Standard (reference [1]) exist. FPGA and ASIC implementations generally allow high data throughput with low hardware resource utilization. This is primarily because the Recommended Standard uses only integer arithmetic. Examples of FPGA implementations can be found in references [13] and [14]; the latter is also offered as a technology-independent IP Core. This implementation can achieve higher than 150 Msamples/sec throughput with less than 10-percent resource utilization when targeting a Xilinx Virtex-5QV FPGA. The Payload Rice Data Compressor (PRDC) (reference [15]) is a radiation-hardened ASIC implantation of the Recommendation with complexity of approximately 70 Kgates. The manufacturing process is TEMIC MG1RT 0.6 mum SCMOS2/2RT with 3 metal layers.

When intending to implement this Recommendation in FPGA or ASIC, several features of the Recommendation can be exploited to reduce resource utilization and increase throughput.

- The maximum size register required to store each block of encoded codewords is equal to the length of the no-compression option; therefore it is not necessary to calculate the register size for all coding options. It is sufficient to set the register buffer size of each block to the length of the no-compression option.
- Appropriate pipelining can yield an implementation that compresses a block in a single clock cycle. To achieve this, a set of FIFOs can be used to store a block of samples to be compressed, and then encoded when options have been determined based on block encoded bit length.
- As shown in reference [16], the encoded length as a function of code option k, denoted L_k, is convex cup in k. Therefore subtracting the length of consecutive options, $L_k - L_{k+1}$, and finding among the results the first negative difference, would identify the option k that minimizes L_k. This makes it possible to calculate the length of all coding options in parallel and avoid unnecessary iterations.
- When the split-sample option is selected for encoding, even though the FS sequence needs to be appended to the codeword prior to the split-sample sequence for a block, the codeword generation does not necessarily require $2 \times J$ iterations. An efficient parallel processing architecture for the FS sequence and storing of intermediate split-sample bits can create the encoded sequence in J iterations for a block.
- Implementation complexity scales up with block size; however, memory requirements are generally low.

Figure 5-4 shows a block diagram of a possible implementation.

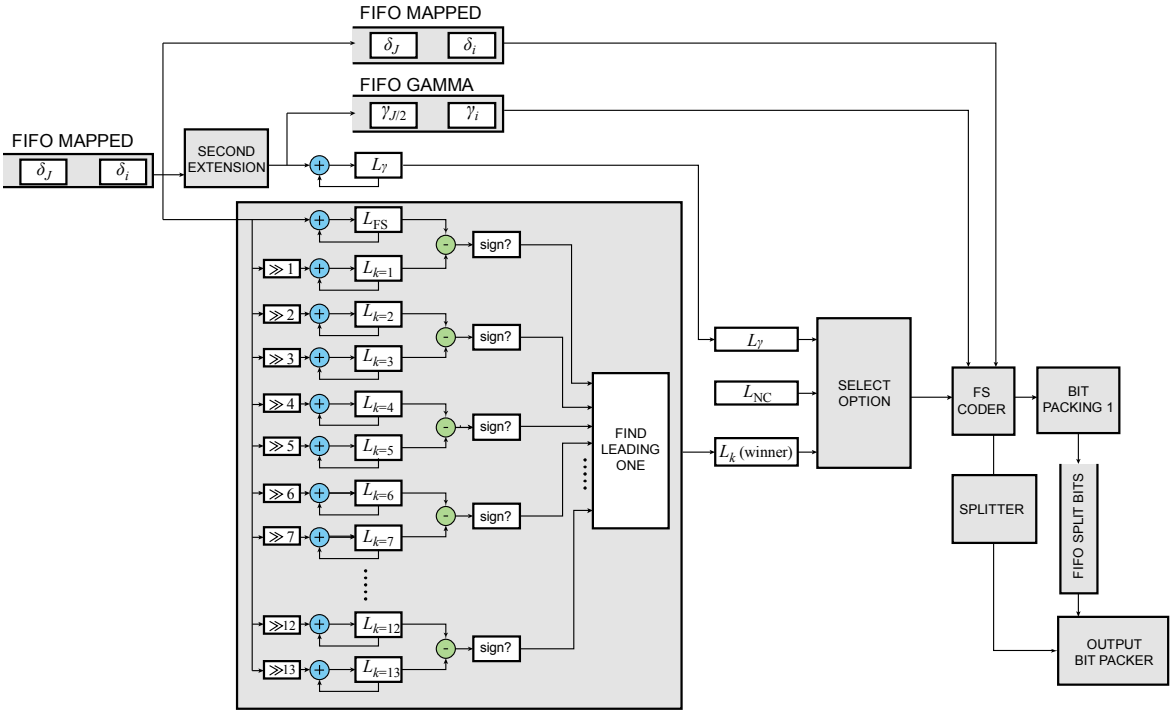


Figure 5-4: Simplified Block Diagram of a Possible Implementation of the Recommendation

5.6 PARALLEL IMPLEMENTATION

The fact that the encoder works in independent blocks makes an implementation very appealing to be done by independent processing units. Nevertheless, the particularity of the Zero-Block option, in which more than one block is encoded, needs to be considered. Additionally, appropriate synchronization is also required, as the CDSes have to be produced in the right order so that the information can be later decompressed. So even if CDSes are produced in parallel, they have to be transmitted in order, and they also have to be sequenced in order in the compressed file.

An example of a parallelization approach is provided in reference [17].

6 PERFORMANCE

6.1 MEASURE

There are two measures of Lossless Data Compression performance: the encoded data rate, measured in bits per sample, and the processing time. The latter will not be addressed in this document, since it is highly dependent upon the processing engine.

The lowest data rate at which a source can be Losslessly represented is referred to as the entropy of the source. That is, a source with entropy of H bits per sample requires an average at least H bits to represent each sample. For example, if outputs from a discrete source are independent and identically distributed with probabilities $p_1, p_2, \ldots p_{(2^n-1)}$, then the source has entropy:

$$H = -\sum_{0}^{2^n-1} p_i \log_2 p_i$$

If the source outputs are not independent, then the entropy depends on higher-order probability distributions. Most real-world sources are sufficiently complex that it is impossible to compute the source entropy exactly, and thus the ultimate compression limit achievable on any source is usually unknown.

It is tempting to measure the first order probability distribution for a source and calculate the 'first order' entropy using the above expression. However, this approach produces an accurate estimate only if the output symbols are independent. By applying various Lossless compression techniques, an upper bound on the entropy of a source can be estimated. It is also possible to develop a probabilistic model of the source and compute the entropy of the model.

The Compression Ratio (CR) is the ratio of the number of bits per sample before compression to the encoded data rate, so for the Lossless compression algorithm:

$$\mathrm{CR} = \frac{n \cdot total\ number\ of\ samples}{\sum length\ of\ the\ encoded\ CDS\ in\ bits},$$

where n is the sample resolution.

Normally for imaging science data, the CR is approximately two. In the worst case, when the No-Compression option is selected, the resulting CR will be slightly less than one because of the ID overhead bits.

6.2 EXAMPLES

The following subsections contain compression-study results, most of which were extracted from reference [5], for eight different scientific imaging and non-imaging instruments. The compression performance, expressed as CR, is summarized in table 6-1. It should be noted that the CR is instrument-data dependent. For a specific instrument, the CR can vary from one data set to the next. The reader should refer to each individual example following table 6-1 for more details on the instrument data set.

Table 6-1: Summary of Data Compression Performance for Different Scientific Data Sets

Paragraph	Instrument Data Set	Compression Ratio
Imaging	a) Thematic Mapper	1.83
	b) Hyperspectral imager (HSI)	2.6
	c) Heat Capacity Mapping Radiometer	2.19
	d) Wide Field Planetary Camera	2.97
	e) Soft X-Ray Solar Telescope	4.69
Non-Imaging	f) Goddard High Resolution Spectrometer	1.72
	g) Acousto-Optical Spectrometer	2.3
	h) Gamma-Ray Spectrometer	5 to 26[1]

[1] depending on integration time

a) **Thematic Mapper (TM)**

Mission Purpose: The Landsat program was initiated for the study of the Earth's surface and resources. Landsat-1, Landsat-2, and Landsat-3 were launched between 1972 and 1978. Landsat-4 and Landsat-5 were launched in 1982 and 1984, respectively.

Landsat Thematic Mapper: on Landsat-4 and 5, the TM data represent typical land observation data acquired by projecting mirror-scanned imagery to solid-state detectors. The sensor for bands 1-5 is a staggered 16-element photo-diode array; it is a staggered 4-element array for band 6. An image acquired on Landsat-4 at 30m ground resolution for band 1 in the wavelength region of 0.45 - 0.52 μm is shown in figure 6-1. This 8-bit 512 x 512 image was taken over Sierra Nevada in California and has relatively high information over the mountainous area.

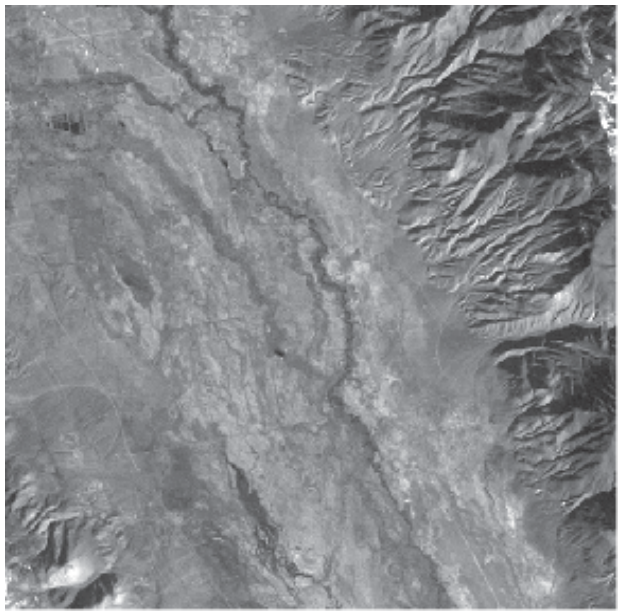


Figure 6-1: Landsat-4 Image

Compression Study: Using a unit-delay predictor in the horizontal direction, setting a block size of 16 samples, and inserting one reference per every image line, the Lossless compression gives a CR of 1.83 for the 8-bit image.

b) **HSI on Small Satellite Technology Infusion Lewis Mission**[1]

Mission Purpose: The Small Satellite Technology Infusion (SSTI) mission objective is to transition to future missions new technologies that have significant measurable performance benefits. The mission will be launched in 1997.

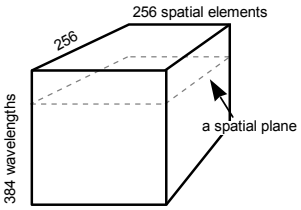


Figure 6-2a: An HSI Data Cube

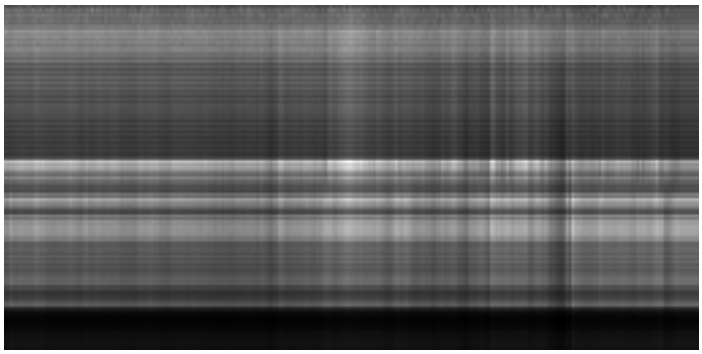


Figure 6-2b: A VNIR Spatial-Spectral Plane

HSI: The instrument contains three CCD arrays of detectors. Two of them are used to collect spectral-spatial information in the wavelength range from 0.4 to 2.5 μm: the Visible Near Infrared (VNIR) imager and the Short-Wave Infrared (SWIR) imager. These imagers provide a 256 pixel spatial line with 128 spectral band information from the VNIR detector and 256 spectral bands from the SWIR detector. Both VNIR and SWIR outputs are quantized to 12-bit data, and the ground resolution is 30m. The third CCD is a panchromatic imager that provides 5m ground resolution at 8-bit output. This CCD is a linear array of 2592 elements. Data compression will be applied only to the VNIR and SWIR data cubes, which are sets of 256 x 256 spatial information with 384 spectral information (see figure 6-2a).

Compression Study: Until actual instrument data can be collected after the launch, the simulation is performed on a data cube simulated from Airborne Visible/Infrared Imaging Spectrometer (AVIRIS) data. Compression is applied to either VNIR or SWIR data separately but in the same manner, that is, along the spatial direction in a spatial-spectral plane shown in figure 6-2b. An average compression ratio of 2.6 was obtained over both the VNIR and SWIR data. If multispectral prediction is applied between the spectral lines, the CR is increased to 3.44. Figure 6-2c shows a spatial plane cut through the data cube.

Figure 6-2c: A VNIR Spatial Plane

[1] The SSTI/Lewis mission failed shortly after launch in 1997, before becoming operational.

c) **Heat Capacity Mapping Radiometer**

Mission Purpose: Launched in 1978, the mission supported exploratory scientific investigations to establish the feasibility of utilizing thermal infrared remote sensor-derived temperature measurements of the Earth's surface within a 12-hour interval, and to apply the night/day temperature-difference measurements to the determination of thermal inertia of the Earth's surface.

Heat Capacity Mapping Radiometer: The sensor is a solid-state photo-detector sensitive in either the visible or the infrared region. A typical 8-bit image taken in the visible light region is given in figure 6-3, at a ground resolution of 500m over the Chesapeake Bay area on the U.S. East Coast.

Compression Study:

One-dimensional prediction: choosing unit-delay prediction in the horizontal direction, and setting block size *J* at 16, Lossless compression gives a CR of 2.19.

Two-dimensional prediction: using a two-dimensional predictor which takes the average of the previous sample and the sample on the previous scan line, and keeping other parameters the same, Lossless compression achieves a CR of 2.28, or about a 5% increase over a one-dimensional predictor.

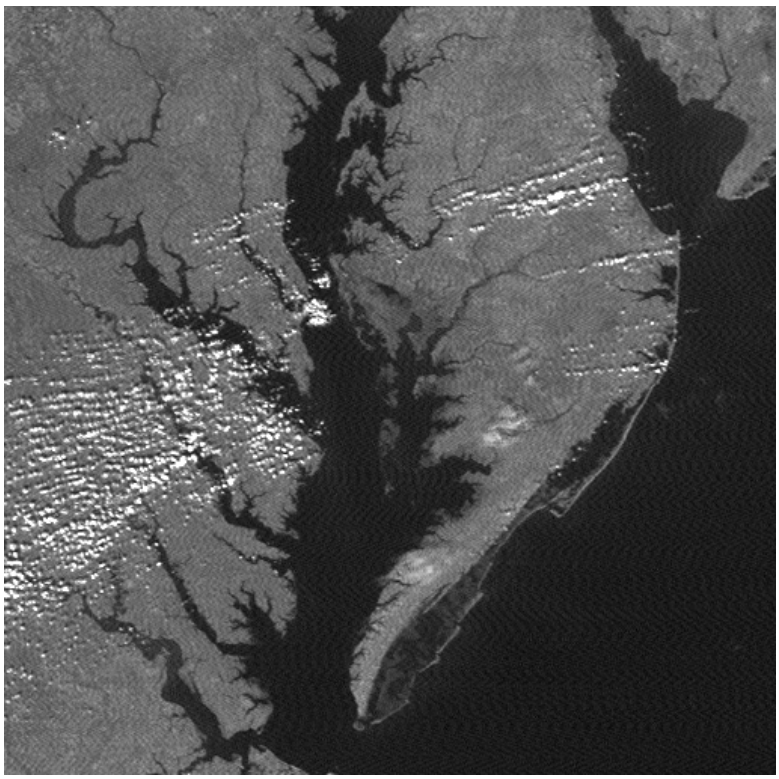


Figure 6-3: Heat Capacity Mapping Radiometer Image

d) **Wide Field Planetary Camera (WFPC) on Hubble Space Telescope**

Mission Purpose: The Hubble Space Telescope (HST), launched in 1990, is aimed at acquiring astronomy data at a resolution never achieved before and observing objects that are 100 times fainter than other observatories have observed.

WFPC: The camera uses four area-array CCD detectors. A typical star field observed by this camera is shown in figure 6-4. This particular image data has maximum value of a 12-bit resolution, which includes background noise of 9 bits.

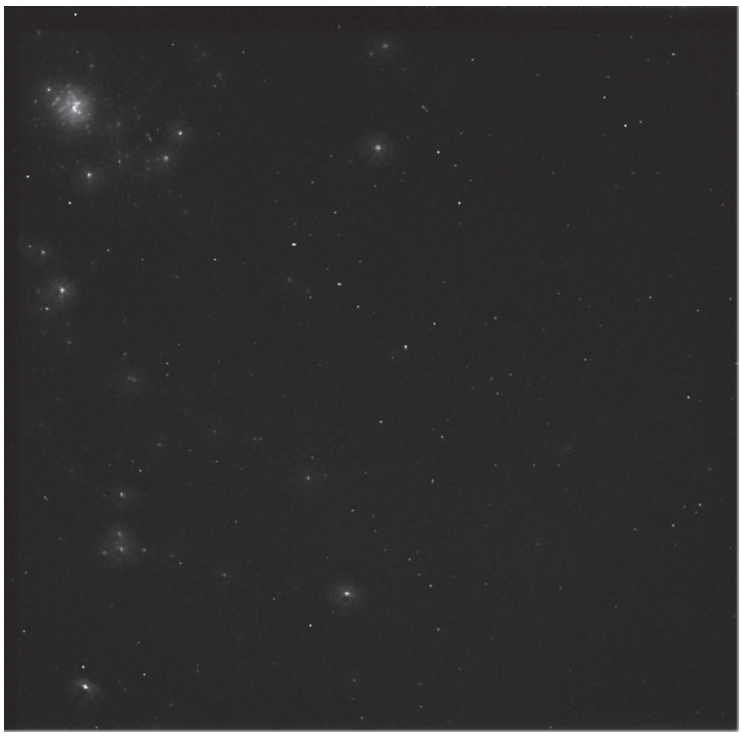


Figure 6-4: Wide Field Planetary Camera Image

Compression Study: Two studies were performed. The first was an investigation into Lossless compression performance on a thresholded star-field at various background noise levels. The second was a study of the effect on the compression ratio produced by varying the length of the source data to be compressed in a packet.

Threshold Study: The effect of thresholding a star field on the compression performance was explored in this study. Different threshold values were applied to this image, and the values of the data were restricted to the threshold value or greater. The thresholding operation did not change the visual quality of the image. All the bright stars were unaffected. The resultant image was compressed Losslessly, and the compression made use of the low-entropy option frequently over the image. The original image in figure 6-4 has a minimum quantized value of 423 and an array size of 800 x 800. The compression was performed using a block size of 16 and a unit-delay predictor in the horizontal line direction. Results are summarized in table 6-2.

Table 6-2: CR at Different Threshold Values

Threshold Values	CCSDS LDC	LZW
None	2.97	3.30
430	2.99	3.33
440	11.92	17.15
450	41.20	63.53
460	53.52	91.48
470	63.66	110.73

In table 6-2, the CCSDS Lossless Data Compression (LDC) CRs for the WFPC study are compared with CRs obtained using the Lempel-Ziv-Welch (LZW) algorithm. Although the performance of the LZW algorithm over the whole image appears better than that of CCSDS LDC algorithm, the results are reversed after considering the error propagation and packetization issues involved in implementing a Lossless algorithm.

Packetization Study: When a packet data format is used for the compressed data, it is expected that during decompression the reconstruction error caused by a single bit error in the packet will not propagate to the next packet, because the data packets are independent of each other and no code tables or statistics of data are passed from one packet to the next. To explore the effect of packet size on the compression performance, several individual lines from figure 6-4 are extracted and compressed separately. The results in table 6-3 are obtained on lines 101 through 104, which cover partially the brightest star cluster on the upper left corner of figure 6-4.

Table 6-3: Compression Ratio vs. Packet Size

Threshold	Packet Size No. of Lines	CCSDS LDC	LZW
None	1 (Line 101)	2.89	1.94
None	1 (Line 102)	2.90	1.92
None	1 (Line 103)	2.86	1.90
None	1 (Line 104)	2.88	1.90
None	2 (Line 101-102)	2.90	2.13
None	2 (Line 103-104)	2.87	2.09
None	4 (Line 101-104)	2.89	2.29
470	1 (Line 101)	16.24	6.98
470	1 (Line 102)	16.92	6.98
470	1 (Line 103)	15.42	6.94
470	1 (Line 104)	16.24	6.67
470	2 (Line 101-102)	16.45	8.70
470	2 (Line 103-104)	15.80	8.60
470	4 (Line 101-104)	16.06	10.30

e) **Soft X-Ray Telescope (SXT) on Solar-A Mission**

Mission Purpose: The Solar-A mission, renamed the Yohkoh mission after its successful launch in August 1991, is dedicated to the study of solar flares, especially of high-energy phenomena observed in the X- and Gamma-ray ranges.

SXT: The SXT instrument is a grazing-incidence reflecting telescope for the detection of X-rays in the wavelength range of 3-60 Angstrom. It uses a 1024 x 1024 CCD detector array to cover the whole solar disk. Data acquired for data compression analysis is shown in figure 6-5.

Compression Study: A unit-delay predictor is applied to the high-contrast image. A CR of 4.69 is achieved, which means that only 3.2 bits are needed per pixel to provide the full precision of the 15-bit image.

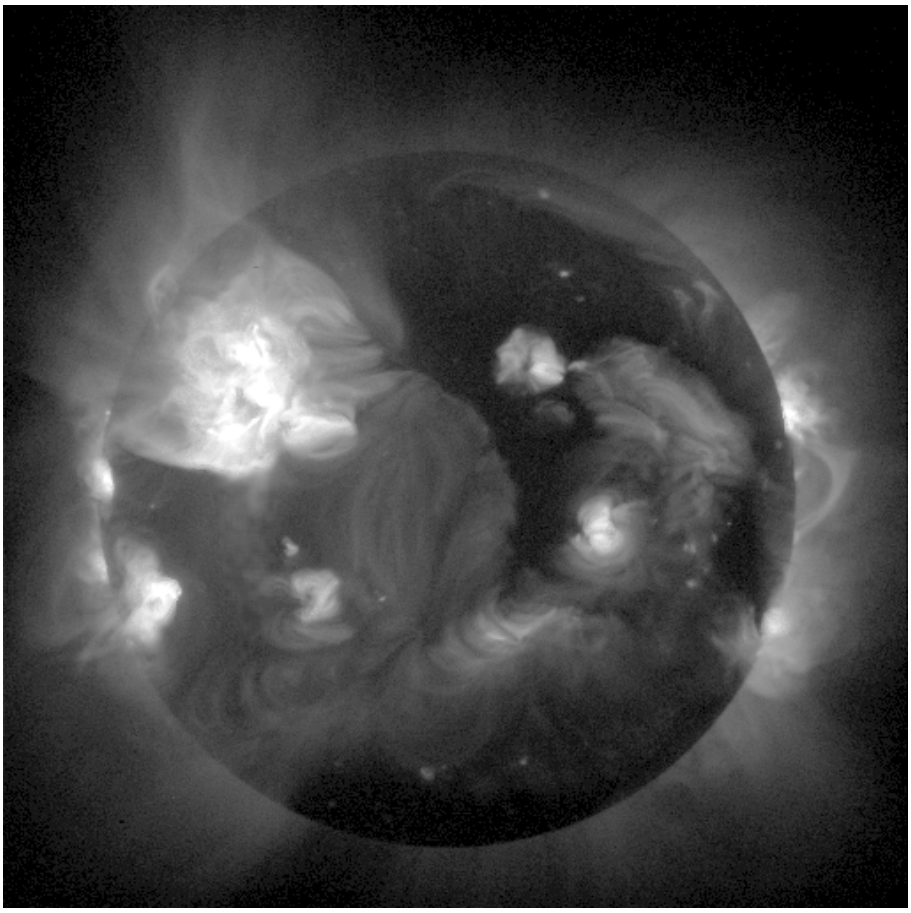


Figure 6-5: Soft X-Ray Telescope Solar Image

f) **Goddard High Resolution Spectrometer (GHRS) on Hubble Space Telescope**

Mission Purpose: The Hubble Space Telescope, launched in 1990, is aimed at acquiring astronomy data at a resolution never achieved before and observing objects that are 100 times fainter than other observatories have observed.

GHRS: The primary scientific objective of the GHRS instrument is to investigate the interstellar medium, stellar winds, evolution, and extra-galactic sources. Its sensors are two photo-diode arrays optimal in different wavelength regions, both in the UV range. Figure 6-6 gives examples of a typical spectrum and background trace. These traces have 512 data values; each is capable of storing digital counts in a dynamic range of 10^7. Spectral shapes usually do not vary greatly. However, subtle variations of the spectrum in such a large dynamic range presents a challenge for any Lossless data compression algorithm.

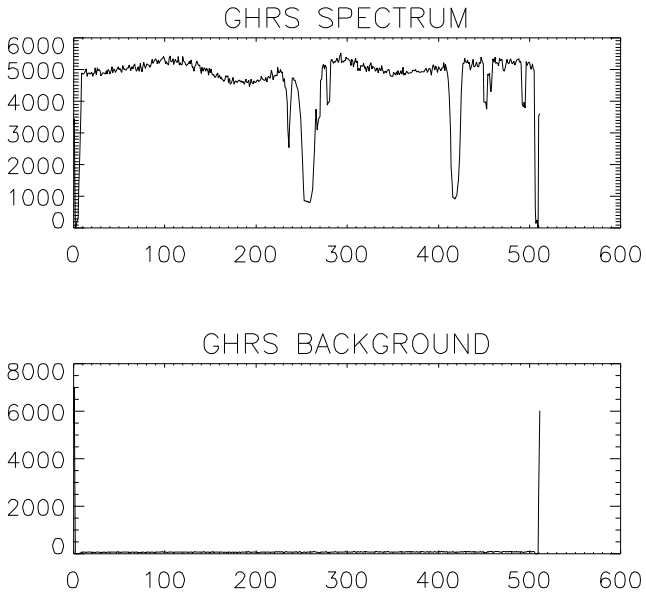


Figure 6-6: Goddard High Resolution Spectrometer

Compression Study: Two different prediction schemes are applied: the first uses the previous sample within the same trace and the second uses the previous trace in the same category (spectrum or background) as a predictor. The results are summarized below in table 6-4. The test data set contains data with a maximum value smaller than the 16-bit dynamic range, and the CR was thus calculated relative to 16 bits.

Table 6-4: Compression Ratio for GHRS

Trace	Predictor	CR
Spectrum 1	Previous Sample	1.64
Spectrum 2	Previous Sample	1.63
Spectrum 2	Spectrum 1	1.72
Background 1	Previous Sample	2.51
Background 2	Previous Sample	2.51
Background 2	Background 1	1.53

g) **Acousto-Optical Spectrometer on Submillimeter Wave Astronomy Satellite (SWAS)**

Mission Purpose: The SWAS is a Small Explorer (SMEX) mission, scheduled for launch in 1998 aboard a Pegasus launcher. The objective of the SWAS is to study the energy balance and physical conditions of the molecular clouds in the galaxy by observing the radio-wave spectrum specific to certain molecules. It will relate these results to theories of star formation and planetary systems. The SMEX platform provides relatively inexpensive and shorter development time for the science mission. SWAS is a pioneering mission that packages a complete radio astronomy observatory into a small payload.

Acousto-Optical Spectrometer: The Acousto-Optical Spectrometer utilizes a Bragg cell to convert the radio frequency energy from the SWAS submillimeter wave receiver into an acoustic wave, which then diffracts a laser beam onto a CCD array. The sensor has 1450 elements with 16-bit readout. A typical spectrum is shown in figure 6-7(a). Expanded view of a portion of two spectral traces is given in figure 6-7(b). Because of the detector non-uniformity, the difference in the ADC gain between even-odd channel, and effects caused by temperature variations, the spectra have non-uniform offset values between traces in addition to the saw-tooth shaped variation between samples within a trace. Because of limited available onboard memory, a CR greater than 2 is required for this mission.

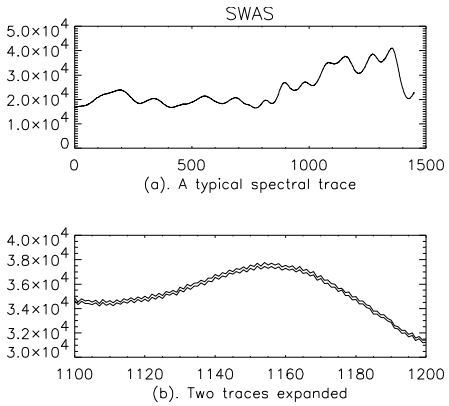


Figure 6-7: Acousto-Optical Spectrometer Waveform

Compression Study: The large dynamic range and the variations within each trace present a challenge to the compression algorithm. Unit-delay prediction between samples is ineffective when the odd and even channels have different ADC gains. Three prediction modes are studied; the results are given in table 6-5.

Table 6-5: SWAS Compression Result

Predictor	CR
Previous Sample	1.58
Previous Trace	1.61
Previous Trace (Multispectral Higher Order Predictor)	2.32

The results show that, even with the similarity in spectral traces, a predictor using an adjacent trace in a direct manner will not improve the compression performance because of the uneven background offset. The multispectral predictor mode is especially effective in dealing with spatially registered multiple data sources with background offsets.

h) **Gamma-Ray Spectrometer on Mars Observer**

Mission Purpose: The Mars Observer was launched in September 1992. The Observer was to collect data through several instruments to help the scientists understand the Martian surface, atmospheric properties, and interactions between the various elements involved.

Gamma-Ray Spectrometer (GRS): The spectrometer uses a high purity germanium detector for gamma rays. The flight spectrum is collected over sixteen-thousand channels, each corresponding to a gamma-ray energy range. The total energy range of a spectrum extends from 0.2 MeV to 10 MeV. Typical spectra for a 5-second and a 50-second collection time are given in figure 6-8. These spectra show the random nature of the counts; some are actually zero over several bits. The spectral-count dynamic range is 8 bits.

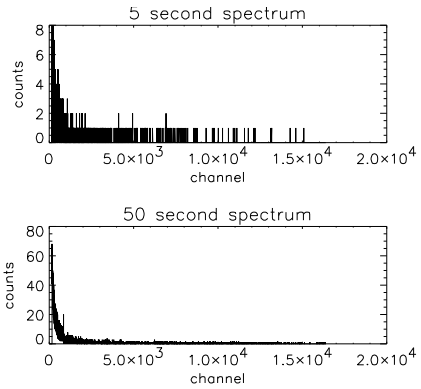


Figure 6-8: GRS Waveform

Compression Study: Two different schemes using the same compression algorithm have been simulated. One scheme applies the Lossless Data Compression algorithm directly to the spectrum, and the other implements a two-layer coding structure.

Direct application of algorithm (one-pass scheme): In the single-pass implementation, the block size J is set at 16, and the entropy-coding mode is selected to bypass the predictor and the mapping function; the algorithm achieves a CR greater than 20 with Lossless compression for the 5-second spectrum. As gamma-ray collection time increases, the achievable compression decreases.

Two-layer scheme: In a two-layer scheme, two passes are needed to compress the data. The first pass finds the channel numbers that have valid counts and generates run-length of channels between them. Meanwhile, a count-file is created that holds only the valid counts as data in the file. In the second pass, both the channel run-length file and count file are compressed using the Lossless compression algorithm at a block size of 16 using the unit-delay prediction mode. The results of both schemes are plotted in figure 6-9. The GRS spectrum offers a specific example that requires an efficient source-coding technique for low-entropy data. For the 5-second spectrum, over 90% of the data are coded with the Zero-Block and Second-Extension options.

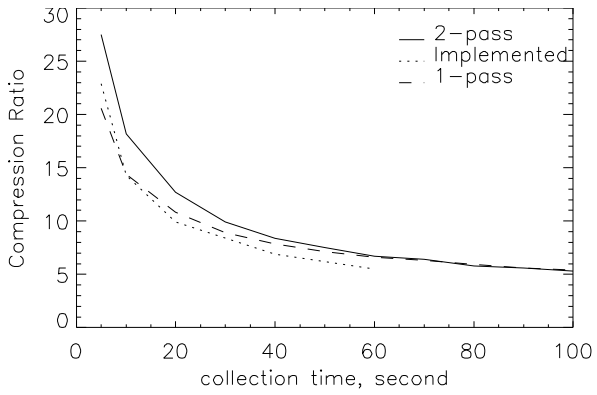


Figure 6-9: Compression Results on GRS Data

ANNEX A

VERIFYING CCSDS LOSSLESS DATA COMPRESSION ENCODER IMPLEMENTATION

A1 PURPOSE

The objective of this annex is to present a limited set of test vectors to verify software or hardware implementations of the CCSDS Data Compression Encoder. This limited set of test vectors can be downloaded from the following location:

http://cwe.ccsds.org/sls/docs/sls-dc/BB121B3TestData

NOTE – The test data are provided 'as is', as a courtesy to users. In no event will the CCSDS, its member Agencies, or the provider of the data be liable for any consequential, incidental, or indirect damages arising out of the use of or inability to use the data.

A2 DESCRIPTION

The directory BB121B3TestData contains files for testing the CCSDS Lossless Data Compression issue 3. The data sets span the full dynamic range from $n = 1$ to 32 and exercise all split-sample options, the no-compression option, the Second-Extension option, and the Zero-Block option at least once. The files use the new File Format defined in issue 3 of the Recommended Standard.

ANNEX B

ACRONYMS AND ABBREVIATIONS

AOS	Advanced Orbiting Systems
APID	application process identifier
AVIRIS	airborne visible/infrared imaging spectrometer
CADU	channel access data unit
CCD	charge-coupled device
CCSDS	Consultative Committee for Space Data Systems
CDS	coded data set
CFDP	CCSDS File Delivery Protocol
CIP	compression identification packet
CR	compression ratio
CVCDU	coded virtual channel data unit
FS	Fundamental Sequence
GHRS	Goddard High Resolution Spectrometer
GRS	gamma-ray spectrometer
HSI	hyperspectral imager
HST	Hubble Space Telescope
LDC	lossless data compression
LSB	least significant bit
LZW	Lempel Ziv Welch
MPDU	multiplexing protocol data unit
NRZ	non-return to zero
PDU	protocol data unit
PRDC	Payload Rice Data Compressor
PSK	phase shift key

RF	radio frequency
ROS	remainder-of-segment
SDU	service data unit
SMEX	Small Explorer
SPDF	Source Packet Data Field
SSTI	small satellite technology infusion
SWAS	Submillimeter Wave Astronomy Satellite
SWIR	short-wave infrared
SXT	Soft X-Ray Telescope
TM	thematic mapper
VCDU	virtual channel data unit
VNIR	visible near infrared
WFPC	Wide Field Planetary Camera