The Consultative Committee for Space Data Systems

Report Concerning Space Data System Standards

LOSSLESS MULTISPECTRAL AND HYPERSPECTRAL IMAGE COMPRESSION

INFORMATIONAL REPORT

CCSDS 120.2-G-1

GREEN BOOK

December 2015

The Consultative Committee for Space Data Systems

Report Concerning Space Data System Standards

LOSSLESS MULTISPECTRAL AND HYPERSPECTRAL IMAGE COMPRESSION

INFORMATIONAL REPORT

CCSDS 120.2-G-1

GREEN BOOK

December 2015

AUTHORITY

Issue:	Informational Report, Issue 1
Date:	December 2015
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
E-mail: secretariat@mailman.ccsds.org

FOREWORD

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 e-mail 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 Federal Science Policy Office (BFSPO)/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.
- Chinese 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.
- Indian Space Research Organization (ISRO)/India.
- Institute of Space Research (IKI)/Russian Federation.
- KFKI Research Institute for Particle & Nuclear Physics (KFKI)/Hungary.
- Korea Aerospace Research Institute (KARI)/Korea.
- Ministry of Communications (MOC)/Israel.
- 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.
- 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 120.2-G-1	Lossless Multispectral and Hyperspectral Image Compression, Informational Report, Issue 1	December 2015	Original issue

CONTENTS

CONTENTS (continued)

CONTENTS (continued)

Figure Page

CONTENTS (continued)

Figure Page

Table

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 Multispectral & Hyperspectral Image Compression* ([1]). Supporting performance information, along with illustrations, is also included. This report provides a broad tutorial overview of the CCSDS Lossless Multispectral & Hyperspectral Image 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 Multispectral & Hyperspectral Image 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 provides an overview of input images, output compressed images, and gives notation used to refer to input image data.
- Section 3 describes the underlying compression algorithm formalized in the Recommended Standard.
- Section 4 examines the impact of different compression settings on compression performance.
- Section 5 discusses practical issues relevant to implementation of the standard.
- Section 6 presents compression performance results for the Recommended Standard and other lossless compression methods.
- Section 7 documents some of the considerations and motivations that influenced the selection of the Recommended Standard and its features, and describes some of the differences between the Recommended Standard and other image compression standards.
- Annex A summarizes the corpus of hyperspectral and multispectral images used for compression testing and evaluation in the course of developing the Recommended Standard.
- Annex B provides links to available software implementations of the Recommended Standard and test data.

- Annex C presents the default compression settings used to derive experimental results.
- Annex D discusses the compressor selection criteria.
- Annex E provides detailed compression results.
- Annex F provides a list of abbreviations and acronyms used in the text of this document.

1.4 CONVENTION

The following convention applies throughout this document:

- The capitalized phrase 'Recommended Standard' by itself refers to *Lossless Multispectral & Hyperspectral Image Compression* (reference [1]).

1.5 TEST IMAGES AND SOFTWARE IMPLEMENTATIONS

Results and examples in this document make use of the set of test images described in annex A. Available software implementations of the Recommended Standard and a synthetic test pattern image are described in annex B.

1.6 REFERENCES

The following documents are referenced in this Report. At the time of publication, the editions indicated were valid. All documents are subject to revision, and users of this Report are encouraged to investigate the possibility of applying the most recent editions of the documents indicated below. The CCSDS Secretariat maintains a register of currently valid CCSDS documents.

[1] *Lossless Multispectral & Hyperspectral Image Compression*. Issue 1. Recommendation for Space Data System Standards (Blue Book), CCSDS 123.0-B-1. Washington, D.C.: CCSDS, May 2012.

[2] M. Klimesh. "Low-Complexity Lossless Compression of Hyperspectral Imagery via Adaptive Filtering." *The Interplanetary Network Progress Report* 42, no. 163 (November 15, 2005): 1–10.

[3] M. Klimesh. "Low-Complexity Adaptive Lossless Compression Of Hyperspectral Imagery." In *Satellite Data Compression, Communications, and Archiving II (August 13–14, 2006, San Diego, California)*, 63000N-1–63000N-9. Edited by R. Heymann, C. Wang, and T. Schmit. SPIE Proceedings vol. 6300. Bellingham, Washington: SPIE, September 2006.

[4] *Space Packet Protocol*. Issue 1. Recommendation for Space Data System Standards (Blue Book), CCSDS 133.0-B-1. Washington, D.C.: CCSDS, September 2003.

[5] *Encapsulation Service*. Issue 2. Recommendation for Space Data System Standards (Blue Book), CCSDS 133.1-B-2. Washington, D.C.: CCSDS, October 2009.

[6] *CCSDS File Delivery Protocol (CFDP)*. Issue 4. Recommendation for Space Data System Standards (Blue Book), CCSDS 727.0-B-4. Washington, D.C.: CCSDS, January 2007.

[7] *AOS Space Data Link Protocol*. Issue 3. Recommendation for Space Data System Standards (Blue Book), CCSDS 732.0-B-3. Washington, D.C.: CCSDS, September 2015.

[8] A. Gersho. "Adaptive Filtering with Binary Reinforcement." *IEEE Transactions on Information Theory* 30, no. 2 (1984): 191–199.

[9] B. Widrow and M. E. Hoff, Jr. "Adaptive Switching Circuits." *IRE WESCON Convention Record* 4 (August 1960): 96-104.

[10] B. Widrow, et al. "Adaptive Noise Cancelling: Principles and Applications." *Proceedings of the IEEE* 63, no. 12 (1975): 1692–1716.

[11] S. Golomb. "Run-Length Encodings (Corresp.)." *IEEE Transactions on Information Theory* 12, no. 3 (July 1966): 399–401.

[12] M.J. Weinberger, G. Seroussi, and G. Sapiro. "The LOCO-I Lossless Image Compression Algorithm: Principles and Standardization into JPEG-LS." *IEEE Transactions on Image Processing* 9, no. 8 (August 2000): 1309–1324.

[13] *Lossless Data Compression*. Issue 2. Recommendation for Space Data System Standards (Blue Book), CCSDS 121.0-B-2. Washington, D.C.: CCSDS, May 2012.

[14] R.G. Gallager and David C. Van Voorhis. "Optimal Source Codes for Geometrically Distributed Integer Alphabets (Corresp.)." *IEEE Transactions on Information Theory* 21, no. 2 (1975): 228–230.

[15] A. Kiely. "Selecting the Golomb Parameter in Rice Coding." *The Interplanetary Network Progress Report* 42, no. 159 (November 15, 2004).

[16] *Lossless Data Compression*. Issue 3. Report Concerning Space Data System Standards (Green Book), CCSDS 120.0-G-3. Washington, D.C.: CCSDS, April 2013.

[17] E. Augé, et al. "Performance Impact of Parameter Tuning on the CCSDS-123 Lossless Multi- and Hyperspectral Image Compression Standard." *SPIE Journal of Applied Remote Sensing* 7, no. 1 (August 26, 2013).

[18] M. Klimesh, A. Kiely, and P. Yeh. "Fast Lossless Compression of Multispectral and Hyperspectral Imagery." In *Proceedings of the International Workshop on On-Board Payload Data Compression (October 28–29, 2010, Toulouse, France)*. Noordwijk, The Netherlands: ESA Conference Bureau, 2010.

[19] *TM Space Data Link Protocol*. Issue 2. Recommendation for Space Data System Standards (Blue Book), CCSDS 132.0-B-2. Washington, D.C.: CCSDS, September 2015.

[20] *TM Synchronization and Channel Coding*. Issue 2. Recommendation for Space Data System Standards (Blue Book), CCSDS 131.0-B-2. Washington, D.C.: CCSDS, August 2011.

[21] *TM Synchronization and Channel Coding—Summary of Concept and Rationale*. Issue 2. Report Concerning Space Data System Standards (Green Book), CCSDS 130.1-G-2. Washington, D.C.: CCSDS, November 2012.

[22] N. Aranki, et al. "Fast and Adaptive Lossless On-Board Hyperspectral Data Compression System for Space Applications." In *Proceedings of the 2009 IEEE Aerospace Conference (March 7–14, 2009, Big Sky, Montana)*, 1–8. New York: IEEE, 2009.

[23] N. Aranki, et al. "Hardware Implementation of Lossless Adaptive and Scalable Hyperspectral Data Compression for Space." In *Proceedings of NASA/ESA Conference on Adaptive Hardware and Systems, 2009 (July 29–August 1, 2009, San Francisco, California)*, 315–322. Piscataway, New Jersey: IEEE Conference Publications, 2009.

[24] N. Aranki, et al. "FPGA Provides Speedy Data Compression for Hyperspectral Imagery." *Xilinx Newsletter* (January 2012).

[25] L. Santos, et al. "HyLoC: A Low-Complexity FPGA Implementation of the CCSDS-123 Standard Algorithm for Multispectral and Hyperspectral Compression." In *Proceedings of the International Workshop on On-Board Payload Data Compression (October 23–24, 2014, Venice, Italy)*. Noordwijk, The Netherlands: ESA Conference Bureau, 2014.

[26] G. Yu, T. Vladimirova, and M.N. Sweeting. "FPGA-Based On-Board Multi/Hyperspectral Image Compression System." *2009 IEEE International Geoscience and Remote Sensing Symposium, IGARSS 2009* 5 (2009): V-212–V-215.

[27] S.R. Tate. "Band Ordering in Lossless Compression of Multispectral Images." *IEEE Transactions on Computers* 46, no. 4 (1997): 477–483.

[28] J-M. Gaucel, et al. "On-Board Compression of Hyperspectral Satellite Data Using Band–Reordering." In *Satellite Data Compression, Communications, and Processing VII (August 23–24, 2011, San Diego, California)*. B. Huang, A.J. Plaza, and C. Thiebaut. SPIE Proceedings vol. 8157. Bellingham, Washington: SPIE, September 2011.

[29] A. Abrardo, et al. "Low-Complexity Approaches for Lossless and Near-Lossless Hyperspectral Image Compression." In *Satellite Data Compression*, 47–66. Edited by B. Huang. New York: Springer, 2011.

[30] *Information Technology—Lossless and Near-Lossless Compression of Continuous-Tone Still Images*. International Standard, ISO/IEC 14495-1:1999. Geneva: ISO, 1999.

[31] *Information Technology—JPEG 2000 Image Coding System: Core Coding System*. 2nd ed. International Standard, ISO/IEC 15444-1:2004. Geneva: ISO, 2004.

[32] I. Blanes and J. Serra-Sagrista. "Pairwise Orthogonal Transform for Spectral Image Coding." *IEEE Transactions on Geoscience and Remote Sensing* 49, no. 3 (2011): 961–972.

[33] *Image Data Compression*. Issue 1. Recommendation for Space Data System Standards (Blue Book), CCSDS 122.0-B-1. Washington, D.C.: CCSDS, November 2005.

[34] *Image Data Compression*. Issue 2. Report Concerning Space Data System Standards (Green Book), CCSDS 120.1-G-2. Washington, D.C.: CCSDS, February 2015.

[35] J. Mielikainen. "Lossless Compression of Hyperspectral Images Using Lookup Tables." *IEEE Signal Processing Letters* 13, no. 3 (2006): 157–160.

[36] M. Slyz and D. Zhang. "A Block-Based Inter-Band Lossless Hyperspectral Image Compressor." In *Proceedings of the Data Compression Conference, DCC 2005 (March 29–31, 2005, Snowbird, Utah)*, 427–436. Piscataway, New Jersey: IEEE Conference Publications, 2005.

[37] I. Gladkova and M. Grossberg. "A Lossless Compression Algorithm for Hyperspectral Data." In *Satellite Data Compression, Communications, and Archiving II (August 13–14, 2006, San Diego, California)*. Edited by R. Heymann, C. Wang, and T. Schmit. SPIE Proceedings vol. 6300. Bellingham, Washington: SPIE, September 2006.

[38] I. Gladkova, M. Grossberg, and S. Gottipati. "Lossless Compression Algorithm for Multispectral Imagers." In *Satellite Data Compression, Communication, and Processing IV (August 10–11, 2008, San Diego, California)*. Edited by B. Huang, R.W. Heymann, and J. Serra-Sagristà. SPIE Proceedings vol. 7084. Bellingham, Washington: SPIE, August 2008.

[39] A.B. Kiely and M.A. Klimesh. "Exploiting Calibration-Induced Artifacts in Lossless Compression of Hyperspectral Imagery." *IEEE Transactions on Geoscience and Remote Sensing* 47, no. 8 (2009): 2672–2678.

[40] E. Christophe, D. Leger, and C. Mailhes. "Quality Criteria Benchmark for Hyperspectral Imagery." *IEEE Transactions on Geoscience and Remote Sensing* 43, no. 9 (2005): 2103–2114.

[41] Shen-En Qian. "Hyperspectral Data Compression Using a Fast Vector Quantization Algorithm." *IEEE Transactions on Geoscience and Remote Sensing* 42, no. 8 (2004): 1791–1798.

[42] Shen-En Qian, et al. "Near Lossless Data Compression Onboard a Hyperspectral Satellite." *IEEE Transactions on Aerospace and Electronic Systems* 42, no. 3 (2006): 851–866.

[43] *Flexible Advanced Coding and Modulation Scheme for High Rate Telemetry Applications*. Issue 1. Recommendation for Space Data System Standards (Blue Book), CCSDS 131.2-B-1. Washington, D.C.: CCSDS, March 2012.

[44] *CCSDS Space Link Protocols over ETSI DVB-S2 Standard*. Issue 1. Recommendation for Space Data System Standards (Blue Book), CCSDS 131.3-B-1. Washington, D.C.: CCSDS, March 2013.

[45] "Packet Version Number." Space Assigned Numbers Authority. http://sanaregistry.org/r/packet_version_number/.

2 OVERVIEW

2.1 INTRODUCTION

The CCSDS *Lossless Multispectral & Hyperspectral Image Compression* Recommended Standard (reference [1]) defines for standardization a particular lossless data compressor that is applicable to three-dimensional images produced by multispectral and hyperspectral imagers and sounders. The Recommended Standard formalizes a version of the 'Fast Lossless' (FL) compressor (see references [2] and [3]) that uses only integer arithmetic.

The compressor is intended to be suitable for use onboard spacecraft; in particular, the algorithm complexity and memory use are designed to be sufficiently low to make high-speed hardware implementation feasible.

In this document it is assumed that the reader is familiar with the contents of reference [1] including terminology defined there.

2.2 INPUT IMAGE

The input to the compressor is an image, which is a three-dimensional array of integer sample values, $s_{z,y,x}$, where x and y are indices in the spatial dimensions and the index z indicates the spectral band. Image sample values may be signed or unsigned. The Recommended Standard supports images with dynamic range (bit depth) between 2 and 16 bits and sizes up to 2^{16} in all three dimensions N_X, N_Y, N_Z.

Image samples produced by multispectral and hyperspectral imagers are typically interleaved in one of three common orderings. In terms of the nesting of the scanning loops, listed from innermost to outermost, the three common orderings are x,y,z (Band-SeQuential [BSQ]), x,y,z (Band-Interleaved Pixels [BIP]), and x,y,z (Band-Interleaved Lines [BIL]). The Recommended Standard is defined in a way that supports all three of these orderings (see 3.3.3).

Certain compression settings tend to be more suitable for certain image types, but the Recommended Standard otherwise makes no distinction between multispectral and hyperspectral images, nor does the compressor make use of the value of the wavelength corresponding to each spectral band in the input image.

While there is not a sharp distinction between multispectral and hyperspectral imagers, generally speaking, hyperspectral imagers typically include hundreds of spectral bands covering a contiguous range of the spectrum, with each band having fairly narrow spectral resolution. A multispectral imager might have tens of bands (or as few as two), the bands might not be contiguous, and each band would generally cover a wider portion of the spectrum. Thus hyperspectral imagery generally exhibits a higher degree of inter-band dependency which can be exploited for compression purposes.

For notational simplicity, both here and in reference [1], data samples and associated quantities may be identified either by reference to the three indices x, y, z, (e.g., $s_{z,y,x}$, $\delta_{z,y,x}$, etc.), or by the pair of indices t, z, (e.g., $s_z(t)$, $\delta_z(t)$, etc.). That is,

$$s_z(t) \equiv s_{z,y,x} \tag{1}$$

$$\delta_z(t) \equiv \delta_{z,y,x} \tag{2}$$

etc., where

$$t = y \cdot N_X + x. \tag{3}$$

The value of t corresponds to the index of a sample within its spectral band when samples in the band are arranged in raster-scan order starting with index t=0. Given t, the values of x and y can be computed as

$$x = t \bmod N_X \tag{4}$$

$$y = \lfloor t / N_X \rfloor. \tag{5}$$

2.3 COMPRESSED IMAGE

The output from the compressor is a compressed image, which is an encoded bitstream from which the input image can be recovered exactly. That is, the Recommended Standard provides lossless compression.

Because of variations in image content, the length of compressed images will vary from image to image. That is, the compressed image is of variable length. Compressed image size also depends on compression settings; section 4 provides further details on these tradeoffs.

A compressed image begins with a variable-length *header* that encodes image and compression parameters followed by a *body* that losslessly encodes the image samples. A compressed image does not include synchronization markers or any other scheme intended to facilitate the automatic identification of the start of a compressed image; it is assumed that the transport mechanism used for the delivery of the compressed image will provide the ability to locate the header of the next image in the event of a bit error or data loss.

In case the encoded bitstream is to be transmitted over a CCSDS space link, several protocols can be used to transfer a compressed image, including:

- Space Packet Protocol (reference [4]);
- CCSDS File Delivery Protocol (CFDP) (reference [6]);
- packet service or bitstream service as provided by the AOS Space Data Link Protocol (reference [7]) or TM Space Data Link Protocol (reference [19]).

Packet Service can carry many types of packets (e.g., Space Packets and Encapsulation Packets) as long as they use a Packet Version Number (PVN) authorized by CCSDS (reference [45]).

Limits on the maximum size data unit that can be transmitted may be imposed by the protocol used or by other practical implementation considerations. The user is expected to take such limits into account when using the Recommended Standard.

Uncorrupted compressed image data are necessary for complete and accurate reconstruction of a compressed image; the effects of a single bit error or loss of compressed image data can propagate to corrupt reconstructed data to the end of a compressed image (see 5.3.1). Therefore measures should be taken to maximize the reliability of the data transmission link. Reference [21] provides information error rates provided by different channel coding options.

In addition, a user may choose to partition the output of an imaging instrument into smaller images that can be independently decompressed, to limit the impact of data loss or corruption on the communications channel and/or to limit the maximum possible size of a compressed image. Under such partitioning, image size can be selected to trade the degree of data protection for compression effectiveness; smaller images provide increased protection against data loss, but tend to reduce overall compression effectiveness. Subsection 5.3.2 provides some examples of this tradeoff.

3 ALGORITHM OVERVIEW

3.1 GENERAL

The Recommended Standard is a formalization of the FL compressor presented in references [2] and [3]; it is an adaptive predictive technique for lossless compression of multispectral and hyperspectral imagery. The FL compressor achieves a combination of low complexity and compression effectiveness that is competitive with the best results from the literature (references [2] and [3]).

The compressor estimates sample values by linear prediction, which is a natural strategy for lossless compression of multispectral and hyperspectral images. The differences between the estimates and the actual sample values are losslessly encoded in the compressed image. Only previously encoded samples are used to predict a given sample so that the prediction operation can be duplicated by the decoder. This is a form of predictive compression, or, more specifically, a form of Differential Pulse Code Modulation (DPCM).

Figure 3-1 depicts the two functional parts of the compressor: a predictor and an encoder.

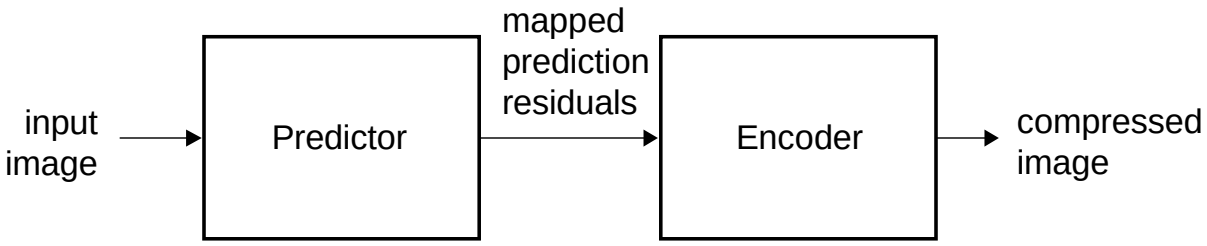


Figure 3-1: Compressor Schematic

The Recommended Standard supports different scan orders, including the common BIL, BIP, and BSQ orderings, for prediction and encoding of samples (see 3.3.3). In all supported scan orders, within any given spectral band sample prediction and encoding are performed in raster scan order. In practice, for a given application a particular scan order choice may be more natural and admit a simpler compressor implementation.

The predictor uses a low-complexity adaptive linear prediction method to predict the value of each image sample based on the values of nearby samples in a small three-dimensional neighborhood. The prediction residual, i.e., the difference between the predicted and actual sample values, is then mapped to an unsigned integer that can be represented using the same number of bits as the input data sample. These mapped prediction residuals make up the predictor output. Subsection 3.2 describes the predictor in more detail.

The compressed image consists of a header that encodes image and compression parameters followed by a body, produced by an entropy coder which losslessly encodes the mapped prediction residuals. Entropy coder parameters are adaptively adjusted during this process to adapt to changes in the statistics of the mapped prediction residuals. Subsection 3.3 describes the entropy coding procedure in more detail.

3.2 PREDICTOR

3.2.1 GENERAL

The underlying FL prediction algorithm on which the Recommended Standard is based is described first. This description uses real-valued quantities and arithmetic. Then described is how this algorithm can be converted to a version that uses only integer arithmetic but is capable of producing essentially equivalent predictions. The Recommended Standard uses this integer-only algorithm.

Prediction can be performed causally in a single pass through the image. The predictor adapts separately for each spectral band, so all scan orders produce the same sample value predictions.

Prediction at sample $s_{z,y,x}$, that is, the calculation of the predicted sample value $\hat{s}_{z,y,x}$ and mapped prediction residual $\delta_{z,y,x}$, defined in equation (35) of reference [1], depends on the values of nearby samples in the current spectral band and P preceding (i.e., lower-indexed) spectral bands, where P is a user-specified parameter. Figure 3-2 illustrates the typical neighborhood of samples used for prediction; this neighborhood is suitably truncated when $y = 0$, $x = 0$, $x = N_X - 1$, or $z < P$.

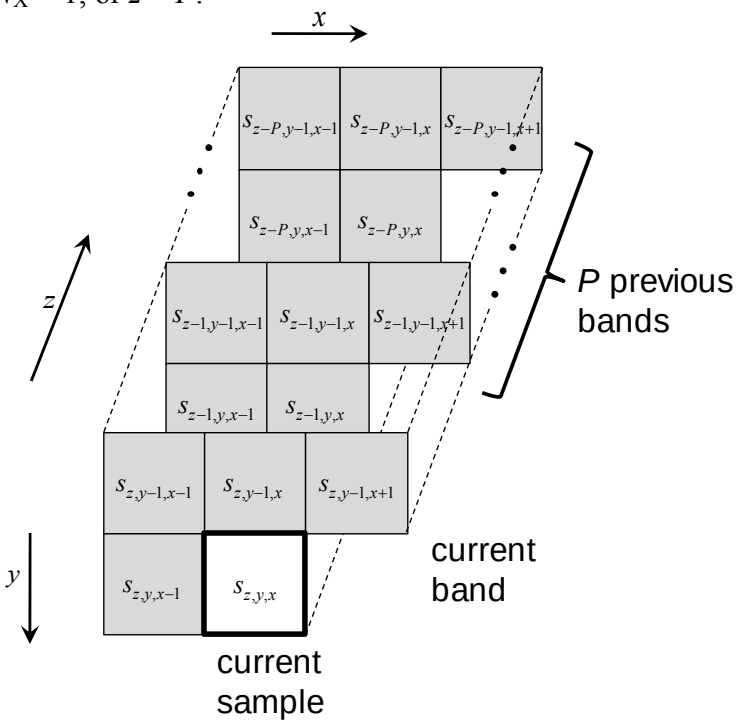


Figure 3-2: Typical Prediction Neighborhood

3.2.2 THE FL PREDICTION ALGORITHM

Within each spectral band z, the FL predictor computes the *local mean* value, $\mu_{z,y,x}$, from previously scanned nearby samples in the spectral band. Figure 3-3 illustrates the samples used to calculate the local mean. A user may choose to use the *neighbor-oriented* local mean, in which case $\mu_{z,y,x}$ is the average of four previously encoded neighboring sample values in the spectral band:

$$\mu_{z,y,x} = \frac{1}{4}\left(s_{z,y,x-1} + s_{z,y-1,x-1} + s_{z,y-1,x} + s_{z,y-1,x+1}\right). \tag{6}$$

If $y = 0$, $x = 0$, or $x = N_X - 1$ then not all of these neighbors exist and $\mu_{z,y,x}$ is suitably modified. It is not necessary to define $\mu_{z,0,0}$. Alternatively, a user may choose to use the *column-oriented* local mean, in which case $\mu_{z,y,x}$ is simply equal to the most recent sample value in the same column:

$$\mu_{z,y,x} = s_{z,y-1,x}, \tag{7}$$

except when $y = 0$, in which case $\mu_{z,0,x} = s_{z,0,x-1}$ (again, $\mu_{z,0,0}$ is not defined).

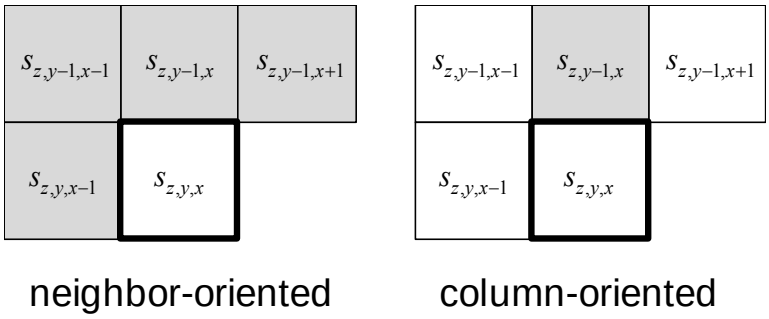


Figure 3-3: Calculating the Local Mean

NOTE – In these diagrams, the local mean $\mu_{z,y,x}$ is equal to the average of the shaded samples.

One can think of the local mean $\mu_{z,y,x}$ as a preliminary estimate of the value of $s_{z,y,x}$. In a sense, the FL algorithm adaptively adjusts prediction weights to predict the amount by which the sample value $s_{z,y,x}$ differs from the preliminary estimate.

Differences between local mean values and previous sample values are arranged in a *local difference* vector, $\Psi_{z,y,x}$. Under *full* prediction mode, the local difference vector is defined as

$$\Psi_{z,y,x} = \begin{bmatrix} s_{z,y-1,x} - \mu_{z,y,x} \\ s_{z,y,x-1} - \mu_{z,y,x} \\ s_{z,y-1,x-1} - \mu_{z,y,x} \\ s_{z-1,y,x} - \mu_{z-1,y,x} \\ s_{z-2,y,x} - \mu_{z-2,y,x} \\ \vdots \\ s_{z-P_z^*,y,x} - \mu_{z-P_z^*,y,x} \end{bmatrix}. \tag{8}$$

Here $P_z^* = \min\{P, z\}$ is the number of preceding (i.e., lower-indexed) spectral bands being used for prediction at band z. Under *reduced* prediction mode, the definition of $\Psi_{z,y,x}$ omits the first three components but is otherwise the same.

The reason for defining two different prediction modes (full and reduced) as well as two different local mean types (column- and neighbor-oriented) is so that the same prediction framework can be used to provide effective prediction for image data from different types of imagers (see 4.2.1).

The first three components of $\Psi_{z,y,x}$ under full prediction mode are called *directional local differences*. Each is equal to the difference between the local mean $\mu_{z,y,x}$ and a previous sample in the same spectral band z. The directional local differences have labels 'N', 'W', and 'NW', suggesting compass directions (see figure 3-4). 'NE', i.e., $s_{z,y-1,x+1} - \mu_{z,y,x}$ is not used. The remaining components of $\Psi_{z,y,x}$ are called *central local differences*. Each is equal to the difference between the sample at the same x and y position in a previous spectral band $z-i$ and the local mean $\mu_{z-i,y,x}$ in that previous spectral band.

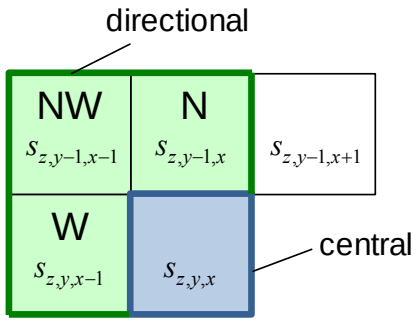


Figure 3-4: Minuends for Computing Directional and Central Local Differences in a Spectral Band

The predicted sample value[1] $\hat{s}^{*}_{z,y,x}$ is equal to the local mean in the current spectral band plus a weighted sum of local difference values from the current and previous spectral bands:

$$\hat{s}^{*}_{z,y,x} = \mu_{z,y,x} + \mathbf{V}_z^{\mathrm{T}}(t)\Psi_{z,y,x} \tag{9}$$

where $\mathbf{V}_z(t)$ is a weight vector having the same dimensions as $\Psi_{z,y,x}$. A separate weight vector is maintained for each band.

Thus the predicted sample value is computed by adjusting the preliminary estimate $\mu_{z,y,x}$ by an increment $\mathbf{V}_z^{\mathrm{T}}(t)\Psi_{z,y,x}$.

Following the calculation of each predicted sample value, the weights used in prediction are adaptively updated using the sign algorithm. The sign algorithm (reference [8]) is a relative of the Least Mean Square (LMS) algorithm (references [9] and [10]), a well-known low-complexity adaptive filtering algorithm. The sign algorithm is also known as the sign-error algorithm and the binary reinforcement algorithm.

Specifically, the prediction error

$$\varepsilon_{z,y,x} = s_{z,y,x} - \hat{s}^{*}_{z,y,x} \tag{10}$$

is used to update the weight vector as follows:

$$\mathbf{V}_z^{\mathrm{T}}(t+1) = \mathbf{V}_z^{\mathrm{T}}(t) + \mathrm{sgn}(\varepsilon_{z,y,x}) \cdot 2^{-\alpha(t)} \cdot \Psi_{z,y,x}. \tag{11}$$

Thus if the predicted value was larger than the true sample value, $\mathrm{sgn}(\varepsilon_{z,y,x})$ will be negative and the weight vector decreases by $2^{-\alpha(t)} \cdot \Psi_{z,y,x}$. Conversely, when the predicted value is less than the true sample value, the weight vector increases by $2^{-\alpha(t)} \cdot \Psi_{z,y,x}$. Here the value of $\alpha(t)$ controls the trade-off between convergence speed and average steady-state error; $\alpha(t)$ begins at some user-specified initial value; then at regular intervals $\alpha(t)$ is increased by one until it reaches some final value. A large value of $\alpha(t)$ (i.e., a small value of the scaling factor $2^{-\alpha(t)}$) results in better steady-state performance but slower convergence.

3.2.3 PREDICTION USING INTEGER ARITHMETIC

The following steps summarize the conversion from the real-valued FL prediction algorithm described above to the integer version used in the Recommended Standard:

– Since the local mean is in general an average of up to four neighboring samples, rather than computing a rational-valued local mean $\mu_{z,y,x}$, the integer predictor computes an integer *local sum* (reference [1]) $\sigma_{z,y,x} = 4\mu_{z,y,x}$.

[1] $\hat{s}^{*}_{z,y,x}$ is a real-valued quantity defined for explanation purposes.

– Similarly, real-valued local differences $\Psi_{z,y,x}$ are scaled by a factor of 4 to produce corresponding integer local differences (reference [1]): $\mathbf{U}_{z,y,x} = 4\Psi_{z,y,x}$.

– To produce an integer weight vector $\mathbf{W}_z(t)$ (reference [1]), real-valued weights $\mathbf{V}_z(t)$ are scaled by a factor of 2^{Ω}, rounded to the nearest integer, and clipped so that each weight component can be represented using Ω+3 bits. Thus $\mathbf{W}_z(t) \approx 2^{\Omega}\mathbf{V}_z(t)$. The corresponding integer round-off and clipping operations are included in the weight update equation (reference [1]). The clipping would be roughly equivalent to constraining real-valued weights to the range [-4, 4]. A larger value of Ω amounts to representing weight components with higher resolution, but these components require more bits to represent; subsection 4.2.4 discusses this tradeoff.

– In updating the weight vector (reference [1]), the integer *weight update scaling exponent* $\rho(t)$ serves the same purpose as (but is not identical to) $\alpha(t)$ in the real-valued weight update equation.

– The Recommended Standard computes the integer *scaled predicted sample value* $\tilde{s}_z(t)$, which is effectively twice the predicted sample value. An equivalently scaled value from the real-valued predictor is

$$2\hat{s}_z^*(t) = 2\left(\mu_z(t) + \mathbf{V}_z^{\mathrm{T}}(t)\Psi_z(t)\right) \approx 2\left(\frac{1}{4}\sigma_z(t) + \frac{1}{2^{\Omega}}\mathbf{W}_z^{\mathrm{T}}(t)\frac{1}{4}\mathbf{U}_z(t)\right) = \frac{2^{\Omega}\sigma_z(t) + \mathbf{W}_z^{\mathrm{T}}(t)\mathbf{U}_z(t)}{2^{\Omega+1}}. \quad (12)$$

The main prediction calculation (reference [1]) rounds this quantity to the nearest integer, takes into account possible register overflow during the calculation (via the mod_R^* operation), and clips the result to account for the range of possible sample values.

– The integer predictor computes the *scaled* prediction error $e_z(t)$, which is effectively twice the prediction error: $2\varepsilon_z(t) = 2\left(s_z(t) - \hat{s}_z^*(t)\right) \approx 2s_z(t) - \tilde{s}_z(t) \equiv e_z(t)$.

The predicted sample value $\hat{s}_z(t)$ is computed from the scaled predicted sample value $\tilde{s}_z(t)$ as (see reference [1])

$$\hat{s}_z(t) = \left\lfloor \frac{\tilde{s}_z(t)}{2} \right\rfloor. \quad (13)$$

The predicted sample value $\hat{s}_z(t)$ is naturally a D-bit quantity, while the integer *scaled* predicted sample value $\tilde{s}_z(t)$, which is effectively twice the predicted sample value, is a (D+1)-bit quantity. Thus $\tilde{s}_z(t)$ retains an extra bit of resolution compared to $\hat{s}_z(t)$. The value of the least significant bit of $\tilde{s}_z(t)$ is discarded in the calculation of $\hat{s}_z(t)$, but the Recommended Standard takes advantage of this extra bit of resolution in the weight update procedure and in the calculation of mapped prediction residuals (reference [1]).

Scaled predicted sample values $\tilde{s}_z(t) = 2\hat{s}_z(t)$ and $\tilde{s}_z(t) = 2\hat{s}_z(t)+1$ both yield the same predicted sample value $\hat{s}_z(t)$. But the smaller value, $\tilde{s}_z(t) = 2\hat{s}_z(t)$, (i.e., an even value of $\tilde{s}_z(t)$) indicates a prediction that $s_z(t) < \hat{s}_z(t)$ is more likely than $s_z(t) > \hat{s}_z(t)$.

As an example, figure 3-5 illustrates the relationship between $\tilde{s}_z(t)$ and $\hat{s}_z(t)$ for an image with unsigned 3-bit samples. If the true sample value is $s_z(t)$ = 5, then the shaded region in the figure corresponds to the range of scaled predicted sample $\tilde{s}_z(t)$ values that are too small, and thus in this region the prediction weights should be increased, while prediction weights should be decreased in the unshaded region. This shaded region corresponds to $e_z(t) \geq 0$, and thus the weight update equation in the Recommended Standard (subsection 4.8.3 of reference [1]) includes a factor of $\text{sgn}^+[e_z(t)]$ rather than $\text{sgn}[e_z(t)]$.

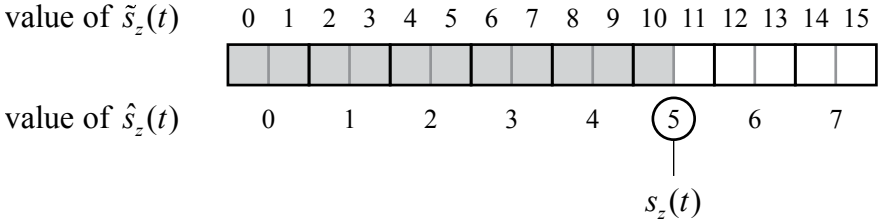


Figure 3-5: Relationship between $\tilde{s}_z(t)$ and $\hat{s}_z(t)$ for an Image with 3-Bit Unsigned Samples

3.2.4 MAPPED PREDICTION RESIDUAL

The prediction residual $\Delta_z(t)$ is the difference between the predicted and actual sample values (reference [1]),

$$\Delta_z(t) = s_z(t) - \hat{s}_z(t). \tag{14}$$

The prediction residual may be positive or negative, but the variable-length codes used by the entropy coder are defined only on nonnegative integer inputs. Thus each prediction residual $\Delta_z(t)$ is mapped to a nonnegative integer $\delta_z(t)$, called the *mapped prediction residual*. This mapping is invertible, so that the decompressor can reconstruct the original sample $s_z(t)$ given $\delta_z(t)$ and $\tilde{s}_z(t)$, and has the property that $\delta_z(t)$ can be represented as a D-bit unsigned integer.

The coding methods available to the entropy coder generally assign longer codewords to larger input values. So for effective compression, the mapping assigns smaller magnitude prediction residuals to smaller mapped values, so that more accurate predictions are more effectively encoded.

The mapping used in the Recommended Standard (reference [1]) is

$$\delta_z(t)=\begin{cases} |\Delta_z(t)|+\theta_z(t), & |\Delta_z(t)|>\theta_z(t) \\ 2|\Delta_z(t)|, & 0\le(-1)^{\tilde{s}_z(t)}\Delta_z(t)\le\theta_z(t) \\ 2|\Delta_z(t)|-1, & \text{otherwise} \end{cases} \tag{15}$$

where $\theta_z(t)=\min\{\hat{s}_z(t)-s_{\min}, s_{\max}-\hat{s}_z(t)\}$. This mapping takes advantage of the extra bit of prediction resolution available from the scaled predicted sample value $\tilde{s}_z(t)$ to produce, on average, smaller mapped prediction residual values.

As an example, figure 3-6 shows the values of the mapped prediction residuals when the predicted sample value is $\hat{s}_z(t)=5$ for an image with unsigned 3-bit samples. The predicted sample value $\hat{s}_z(t)=5$ could have been produced by either $\tilde{s}_z(t)=10$ (shown in blue) or $\tilde{s}_z(t)=11$ (in green). The former case indicates a prediction that $\Delta_z(t)<0$ is more likely than $\Delta_z(t)>0$, and mapped prediction residuals are assigned accordingly.

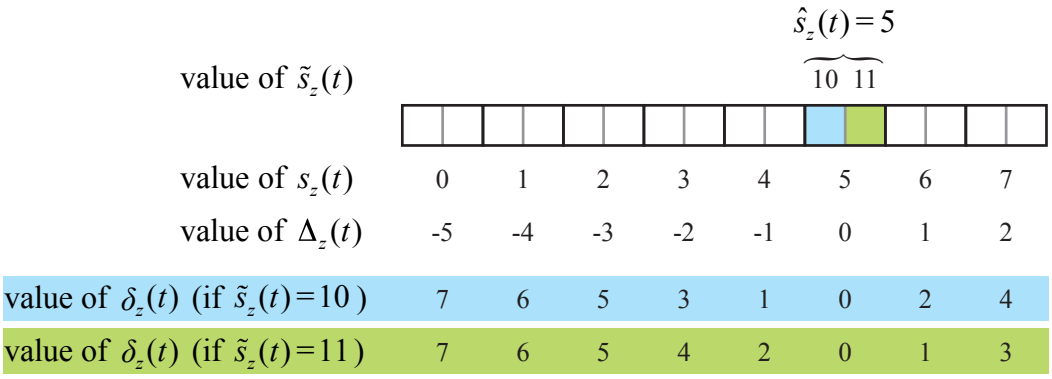


Figure 3-6: Example of Mapped Prediction Residual Values for an Image with 3-Bit Unsigned Samples

Since prediction is causal, the decompressor can use previously decoded sample values to compute $\tilde{s}_z(t)$, which can then be used to compute $\hat{s}_z(t)$ and $\theta_z(t)$. After decoding the value of $\delta_z(t)$ from the compressed bitstream, the prediction residual can be computed by inverting equation (15), which yields

$$\Delta_z(t)=\begin{cases} (\theta_z(t)-\delta_z(t))\,\mathrm{sgn}^{+}(\hat{s}_z(t)-s_{\text{mid}}), & \delta_z(t)>2\theta_z(t) \\ \left\lfloor\dfrac{\delta_z(t)+1}{2}\right\rfloor(-1)^{\tilde{s}_z(t)+\delta_z(t)}, & \delta_z(t)\le 2\theta_z(t). \end{cases} \tag{16}$$

Finally, from equation (14), the sample value $s_z(t)$ can be computed as

$$s_z(t)=\Delta_z(t)+\hat{s}_z(t).$$

3.2.5 WORD SIZES

Table 3-1 lists bounds on the range of possible values, and the corresponding sizes of binary words that could be used to store these quantities, for several key quantities used in prediction. Intermediate calculations necessary to compute some of these quantities may require higher bit depths.

Table 3-1: Bounds and Word Sizes for Predictor Quantities

Symbol	Meaning	Bounds	Bits to Represent
$s_z(t)$	image data sample	$[s_{\min}, s_{\max}]$	D
$\sigma_z(t)$	local sum	$[4s_{\min}, 4s_{\max}]$	$D+2$
$d_z(t)$	central local difference	$\pm 4(2^D-1)$	$D+3$
$d_z^{\mathrm{N}}(t)$, $d_z^{\mathrm{W}}(t)$, $d_z^{\mathrm{NW}}(t)$	directional local differences	$\pm 3(2^D-1)$	$D+3$
$\omega_z^{\mathrm{N}}(t)$, $\omega_z^{\mathrm{W}}(t)$, $\omega_z^{\mathrm{NW}}(t)$, $\omega_z^{(i)}(t)$	weight values	$[-2^{\Omega+2}, 2^{\Omega+2}-1]$	$\Omega+3$
$\tilde{s}_z(t)$	scaled predicted sample value	$[2s_{\min}, 2s_{\max}+1]$	$D+1$
$\hat{d}_z(t)$	predicted central local difference	full mode: $\pm(4P+9)2^{\Omega+2}(2^D-1)$ reduced mode: $\pm(4P\cdot 2^{\Omega+2}(2^D-1)$	full mode: $\Omega+3+\left\lceil \log_2\left\lfloor (4P+9)\left(2^D-1\right)\right\rfloor\right\rceil$ reduced mode: $\Omega+5+\left\lceil \log_2\left\lfloor P\left(2^D-1\right)\right\rfloor\right\rceil$
$\hat{s}_z(t)$	predicted sample value	$[s_{\min}, s_{\max}]$	D
$e_z(t)$	scaled prediction error	$\left\lfloor -2^{D+1}+1, 2^{D+1}-2\right\rfloor$	$D+2$
$\Delta_z(t)$	prediction residual	$\pm\left(2^D-1\right)$	$D+1$
$\delta_z(t)$	mapped prediction residual	$\left\lfloor 0, 2^D-1\right\rfloor$	D

Subsection 4.2.5 discusses the register size parameter R.

3.3 ENCODER

3.3.1 GENERAL

The encoder losslessly encodes the mapped prediction residuals produced by the predictor, creating a compressed image which consists of a *header* followed by a *body*. The variable-length header encodes image and compression parameters. The body consists of losslessly encoded mapped prediction residuals $\{\delta_{z,y,x}\}$ from the predictor.

The original FL algorithm uses an adaptive coding approach using Golomb-Power-Of-2 (GPO2) codes (reference [11]), similar to the approach used in the JPEG-LS image compression standard (reference [12]). The *sample-adaptive* entropy coder specified in the Recommended Standard formalizes a version of this encoder. The Recommended Standard alternatively allows the use of a *block-adaptive* entropy coder. The block-adaptive coder, which also makes use of GPO2 codes, is the Rice coding algorithm as specified in the CCSDS 121.0-B-2 Recommended Standard (reference [13]).

Under the sample-adaptive entropy coding approach, each mapped prediction residual is encoded using a variable-length binary codeword. The variable-length codes used are adaptively selected based on statistics that are updated after each sample is encoded. Separate statistics are maintained for each spectral band, and consequently, the sample-adaptive entropy coder produces the same compressed image size regardless of the order in which samples are encoded. It also tends to provide slightly more effective compression than the block-adaptive coder.

The block-adaptive encoding approach was included as an option in the Recommended Standard so that implementers could take advantage of existing space-qualified hardware implementations of this encoder. Under the block-adaptive entropy coding approach, the sequence of mapped prediction residuals is partitioned into short blocks, and the encoding method used is independently and adaptively selected for each block. Depending on the encoding order, the mapped prediction residuals in a block may be from the same or different spectral bands, and thus the compressed image size depends on the encoding order when this method is used.

Given the knowledge of image and compression parameters, and assuming no corruption of data, the decoder can determine when it has finished decompressing an image. Consequently, the Recommended Standard does not explicitly indicate the compressed image size in the header or use a terminating sequence to mark the end of the compressed image.

3.3.2 HEADER

The header includes fields for all compression settings needed to reconstruct the compressed image. There are only two optional header fields that may be omitted, and thus there are only two scenarios in which information not included in the header might be needed by the decompressor:

a) A user-defined custom weight initialization table might be used by the predictor, and thus needed for decompression, but not encoded in the header.

b) When the sample-adaptive entropy coder is used, a user-defined accumulator initialization table might be used by the encoder, and thus needed for decompression, but not encoded in the header.

Omitting such information might be a logical choice, e.g., if a mission used the same fixed custom weight initialization and/or accumulator initialization throughout a mission. In this case, the mission might choose to reduce compressed image size by omitting this repetitive information from the image header.

3.3.3 ENCODING ORDER

The mapped prediction residuals are sequentially encoded in the compressed image body in the order selected by the user and indicated in the header (see subsection 5.4.2 of reference [1]). The Recommended Standard specifies the allowed orders in which encoded sample values may appear in a compressed image, but this encoding order need not correspond to the order in which samples are output from the imaging instrument or processed by the predictor.

In addition to BSQ encoding order, the Recommended Standard also allows Band-Interleaved (BI) encoding order, which includes BIL and BIP as special cases. Under BI encoding order, a *frame*, defined as the set of all sample values with the same y coordinate (see subsection 5.4.2.2.1 of reference [1]), is partitioned into separate *sub-frames*, consisting of M spectral bands each, except possibly the last sub-frame in a frame. Within each sub-frame, encoding proceeds in BIP order. Figure 3-7 illustrates the BI encoding order for samples in a frame when $M = 3$.

BIL and BIP encoding orders correspond to $M = 1$ and $M = N_Z$, respectively. Other values of M may simplify hardware pipelining and thus facilitate faster compressor implementations. Specifically, prediction for the current sample cannot be performed until the weight vector has been updated using the prediction for the previous sample in the same band. This constraint makes it more difficult to perform pipelining or parallelization under BIL ordering. This is not an issue for BIP processing, because each spectral band has its own weight vector. Under BI ordering, arranging samples into sub-frames permits processing to be performed in BIP order within a sub-frame, thus allowing pipelining.

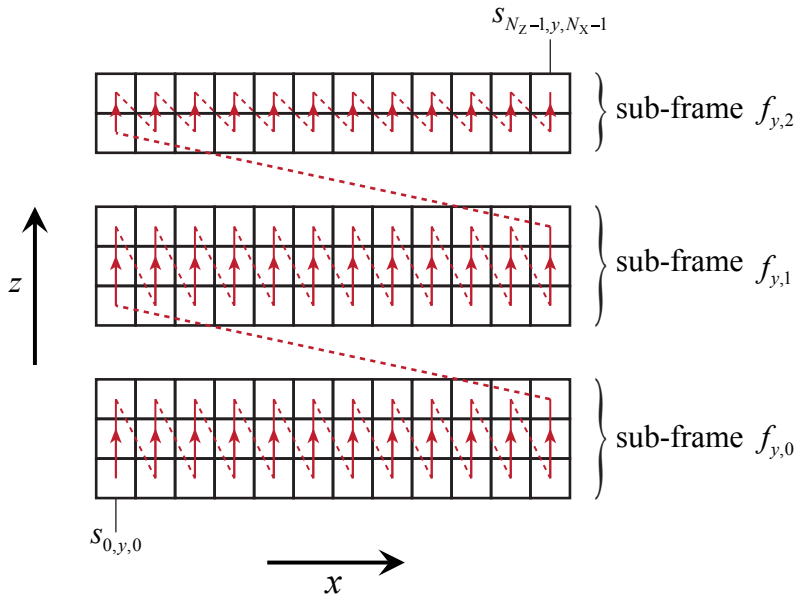


Figure 3-7: Illustration of Sample Encoding Order within a Frame under Band Interleaved Encoding Order for Sub-Frame Interleaving Depth *M*=3

Because the predictor adapts separately for each spectral band, all scan orders produce the same sample value predictions. Under the sample-adaptive entropy coding option, separate entropy coding statistics are maintained for each spectral band, and thus compressed image size does not depend on the order in which samples are encoded. However, as discussed in 4.3.3.2, under the block-adaptive entropy coding option, compressed image size will vary somewhat depending on the sample encoding order, and for some imagers BIL or BSQ encoding order provides a noticeable improvement in compression effectiveness over BIP.

3.3.4 SAMPLE-ADAPTIVE ENTROPY CODER

The variable length codes used by the sample-adaptive entropy coder are based on GPO2 codes (i.e., Golomb codes, references [11] and [14], with parameters that are powers of 2, also known as Golomb-Rice codes). The overall encoding procedure, including Golomb code parameter selection, is very similar to that used by LOCO-I/JPEG-LS, described in reference [12].

The sample-adaptive coder uses length-limited GPO2 codes, constraining the codes so that the maximum codeword length is $U_{\max}+D$ bits. This length limit makes hardware implementation simpler and reduces the cost of encoding occasional outlier samples.

For a given value of the *unary length limit* $U_{\max}$, a family of codes are available, parameterized by the nonnegative integer $k_z(t) \leq D-2$. Each code is a mapping from nonnegative integers to variable-length prefix-free binary codewords.

To encode nonnegative integer δ using the k^{th} length-limited GPO2 code, δ can be written as

$$\delta = u \cdot 2^k + r \tag{17}$$

where u and r are the quotient and remainder when δ is divided by 2^k, that is, $u = \lfloor \delta / 2^k \rfloor$, and $r = \delta \bmod 2^k$.

The codeword for δ depends on whether u is less than $U_{\max}$. If $u < U_{\max}$ then the codeword for δ consists of the unary encoding of u (that is, u zeros followed by a 1) followed by the k-bit binary representation of r, which is simply the k least significant bits of the binary representation of δ. Otherwise $u \geq U_{\max}$ and the codeword for δ consists of $U_{\max}$ zeros followed by the D-bit binary representation of δ. Hence, $U_{\max}$ is called the unary length limit, because it constrains the length of the unary part of the codeword.

To determine which code to use (i.e., the value of k) to encode a mapped prediction residual, the sample-adaptive encoder maintains a running sum $\Sigma_z(t)$ of mapped prediction residuals in the spectral band, called the accumulator, and a counter $\Gamma(t)$. The ratio $\Sigma_z(t)$ / $\Gamma(t)$ is an estimate of the mean mapped prediction residual value, and this estimate is used to select the code parameter k. Specifically, k is the largest nonnegative integer $k \leq D - 2$ satisfying

$$2^k \leq \frac{\Sigma_z(t)}{\Gamma(t)} + \frac{49}{128}, \tag{18}$$

and $k = D - 2$ when this relation is not satisfied by any $k < D - 2$ (see reference [15] for an analysis).

The counter and accumulator are each periodically halved (see subsection 5.4.3.2.2.4 of reference [1]) so that more recent sample values have more impact on the estimated mean value.

The counter and accumulator values used to calculate the coding parameter k are calculated in a causal manner based on previously encoded samples, so the decompressor can determine the value of the coding parameter k for each sample. Given the value of k, to decode the value of δ from the compressed bitstream:

- If the next $U_{\max}$ bits are all zeros, then δ is read from the D bits following the string of $U_{\max}$ zeros.
- Otherwise, the number of consecutive zeros encodes the value of u, and the next k bits encodes the value of r. Then equation (17) can be used to reconstruct the value of δ.

3.3.5 BLOCK-ADAPTIVE ENTROPY CODER

Reference [16] presents a thorough discussion of the CCSDS 121.0-B-2 Recommended Standard (reference [13]) used as the block-adaptive entropy coding option.

3.3.6 WORD SIZES

Table 3-2 lists bounds on the range of possible values, and the corresponding sizes of binary words that could be used to store these quantities, for several key quantities used in sample-adaptive entropy coding.

Table 3-2: Bounds and Word Sizes for Sample-Adaptive Entropy Coder Quantities

Symbol	Meaning	Bounds	Bits to Represent
$\Sigma_z(t)$	accumulator	$[0,2^{D+\gamma^*}-1]$	$D+\gamma^*$
$\Gamma(t)$	counter	$[0,2^{\gamma^*}-1]$	γ^*

4 COMPRESSION SETTINGS

4.1 INTRODUCTION

This section examines the influence of different compression settings on performance. Experimental results, some of which were originally presented in reference [17], are included to illustrate some of the tradeoffs involved. These experiments use the corpus of multispectral and hyperspectral images described in annex A. To avoid a combinatorial explosion in the number of experiments to be performed, in each experiment, one or more interrelated settings are varied while the remaining settings are fixed; in each case the fixed settings are set to the values indicated in the tables of annex C.

For a given source image and choice of compression settings, compression performance is determined by the *compressed data rate* achieved, measured in bits/sample, which is defined as the number of bits used in the compressed representation of the image divided by the number of samples in the image, $N_X \cdot N_Y \cdot N_Z$. Lower values of compressed data rate indicate better compression performance.

4.2 PREDICTOR

4.2.1 PREDICTION MODE AND LOCAL SUM TYPE

The Recommended Standard defines two different prediction modes (full and reduced) and two different local sum types (column- and neighbor-oriented) so that the same prediction framework can be used to provide effective prediction for image data from different types of imagers.

Specifically, the use of column-oriented local sums and reduced mode is intended to provide more effective compression for images that exhibit significant streaking artifacts parallel to the y direction[2] (reference [18]). Such streaking-artifacts are often evident in raw images from pushbroom imagers.

Pushbroom imagers use a two-dimensional detector array to acquire data in spatial-spectral slices. Thus each detector element corresponds to a specific spectral band and cross-track position. Because the characteristics of detector elements generally vary somewhat from element to element, cross-track adjacent samples in a given spectral band will not be as similar as they would be in an instrument that uses the same detector for all samples in a given spectral band (e.g., in a whiskbroom instrument). On the other hand, along-track adjacent samples in the same band will tend to be very similar. As an example, figure 4-1 shows the mean Digital Number (DN) value of each detector element of the Moon Mineralogy Mapper (M3) hyperspectral imager, averaged over several imaging frames.

[2] The corpus includes images from the MODIS instrument, which is in fact a 'whisk-push' imager that exhibits streaking artifacts parallel to the cross-track direction. Thus, following the recommendation in NOTE 1 of subsection 3.2.1 of reference [1], in experiments each band of MODIS images is transposed prior to compression, so that the predominant streaks appear in the y direction.

Some detector elements tend to produce noticeably larger DN values than their neighbors, which produces streaking artifacts parallel to the along-track direction, as evident in the false-color image shown in figure 4-2.

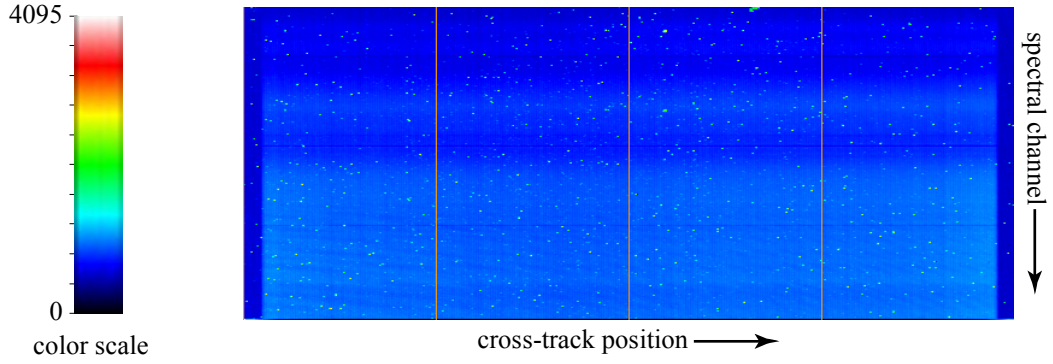


Figure 4-1: Mean DN Value of M3 Detector Array, Using Color Scale at Left, Averaged over 7000 Imaging Frames

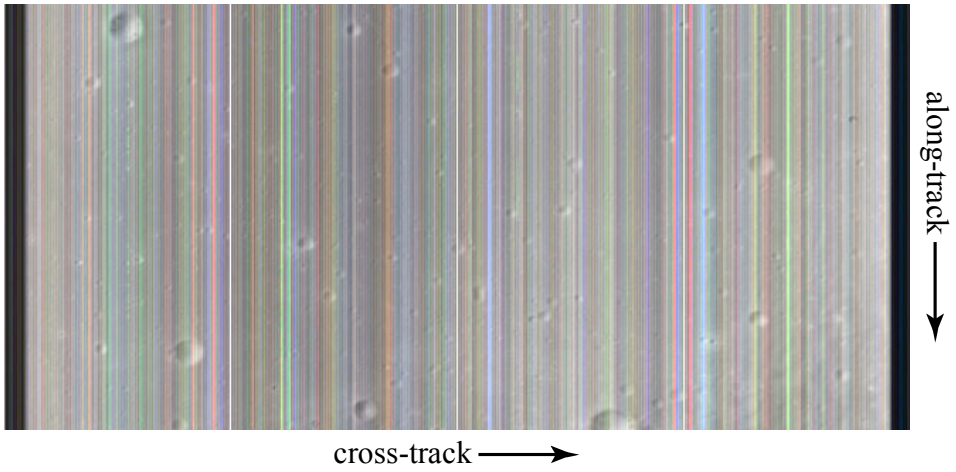


Figure 4-2: False-Color Image Derived from Spectral Channels 200, 201, 202 from a Portion of an M3 Image

For images without such artifacts, such as calibrated images or imagery from whiskbroom instruments, compression effectiveness is generally improved by using neighbor-oriented local sums. Similarly, the use of a pre-processing stage to reduce the severity of streaking artifacts, in combination with neighbor-oriented local sums, may provide more effective compression than obtained on the original image under either choice of local sum (see 5.5).

Figure 4-3 shows compressed data rate as a function of the number of prediction bands, *P*, for six different imagers under all four combinations of prediction mode and local sum type. The relative performance of these four choices generally follows what one would expect based on the presence or absence of streaking artifacts in the input image. On the CRISM FRT, Hyperion raw, and MODIS night images, which exhibit streaking artifacts ranging from moderate to severe, column-oriented local sums give the best performance, and in the case of CRISM FRT images, the use of reduced mode provides a noticeable improvement over full mode. On the AIRS, AVIRIS, and LANDSAT images, which do not exhibit streaking artifacts, neighbor-oriented local sums outperform column-oriented local sums and the choice of full or reduced mode has a relatively small impact on performance.

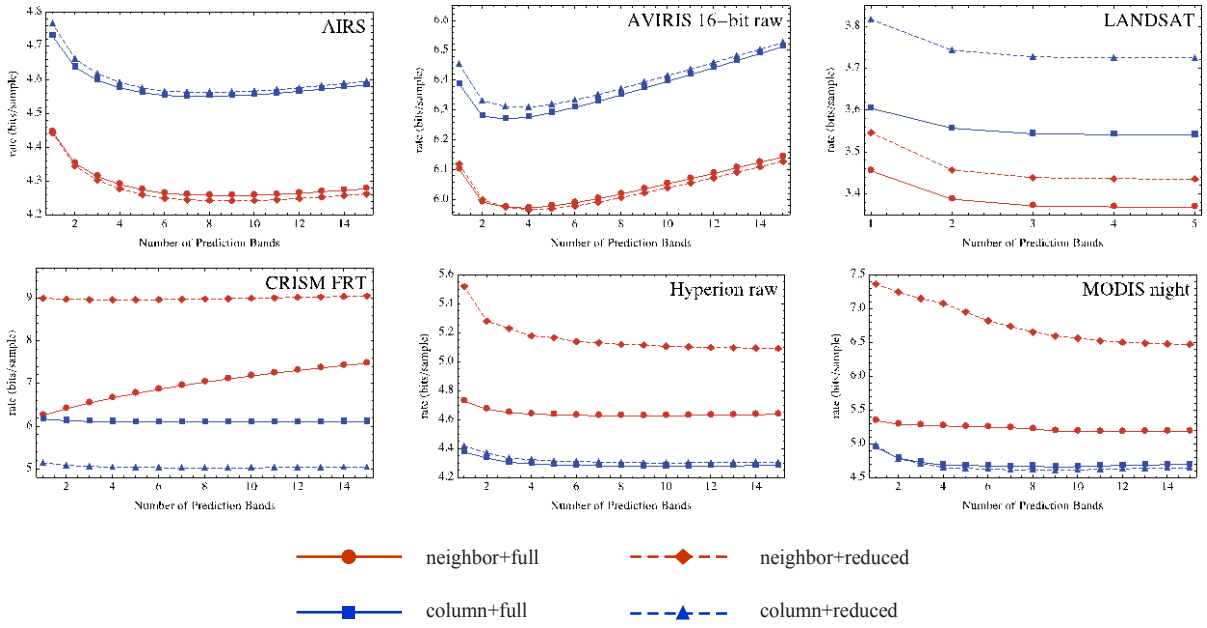


Figure 4-3: Compressed Bit Rate for Different Choices of Prediction Mode and Local Sum Type

NOTE – Results show averages over all sample images of a given type.

For a given value of *P*, the use of full prediction mode requires three additional components in the weight vector compared to reduced mode. Thus slower adaptation of this longer weight vector would be expected, and so for a given image it could be the case that full prediction mode provides a benefit over reduced mode, but only after processing a sufficient number of samples. However, for the images and compression settings used in these experiments, it appears that this effect is typically not significant.

4.2.2 NUMBER OF BANDS FOR PREDICTION

Figure 4-4 shows the compressed data rate for different imagers as a function of the number of previous bands used in prediction, P. As might be expected, setting P=0 (i.e., not using any previous bands for prediction) yields the worst results, often by a dramatic amount. However, diminishing marginal returns are generally observed as P increases; the bit rate curve tends to flatten, or in the case of AVIRIS and IASI, visibly increases. For the images in the corpus, evidently there is not much motivation to use values of P near the maximum allowed, 15.

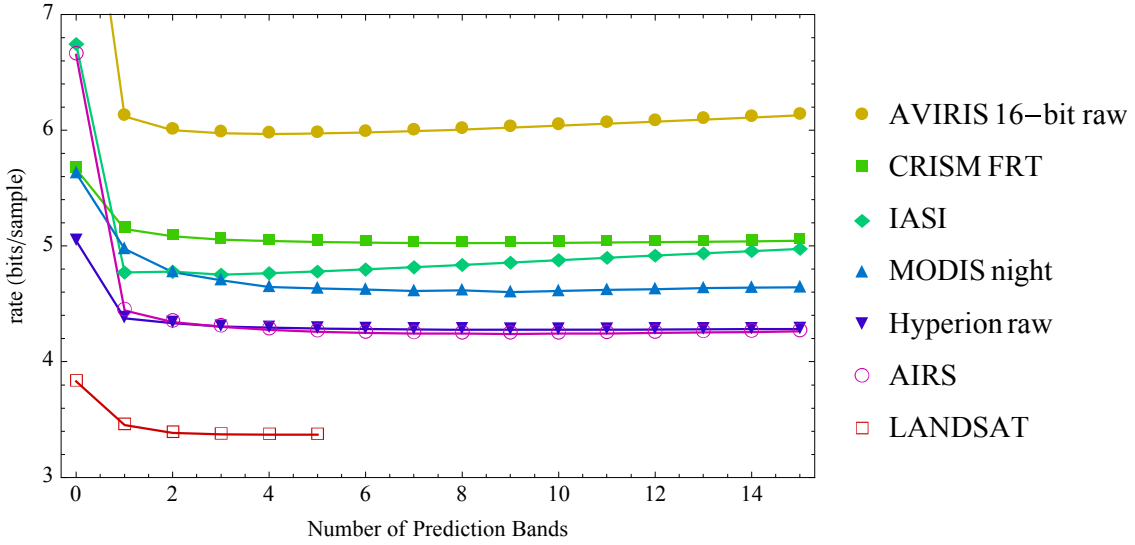


Figure 4-4: Average Compressed Bit Rate Performance as a Function of *P*

4.2.3 WEIGHT ADAPTATION PARAMETERS

Compression parameters v_{min}, v_{max}, and t_{inc} affect the rate at which the predictor adapts to the image. Figure 4-5 shows compressed data rate as a function of v_{max} when v_{min}=−6 and $t_{inc} = 2^7$ and figure 4-6 shows the rate for several different combinations of t_{inc} and v_{min}. It is observed that compression effectiveness can suffer significantly when v_{max} is too small. As v_{max} increases, compression performance improves up to a point, then worsens, but to a much smaller degree.

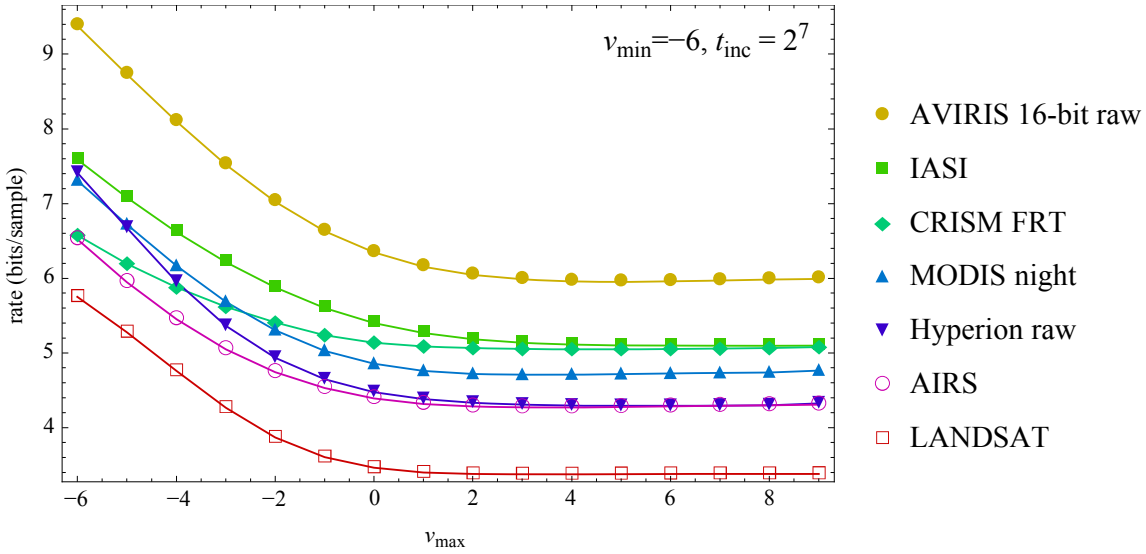


Figure 4-5: Average Compressed Data Rate as a Function of v_{max} When v_{min}=-6 and t_{inc}=2^7

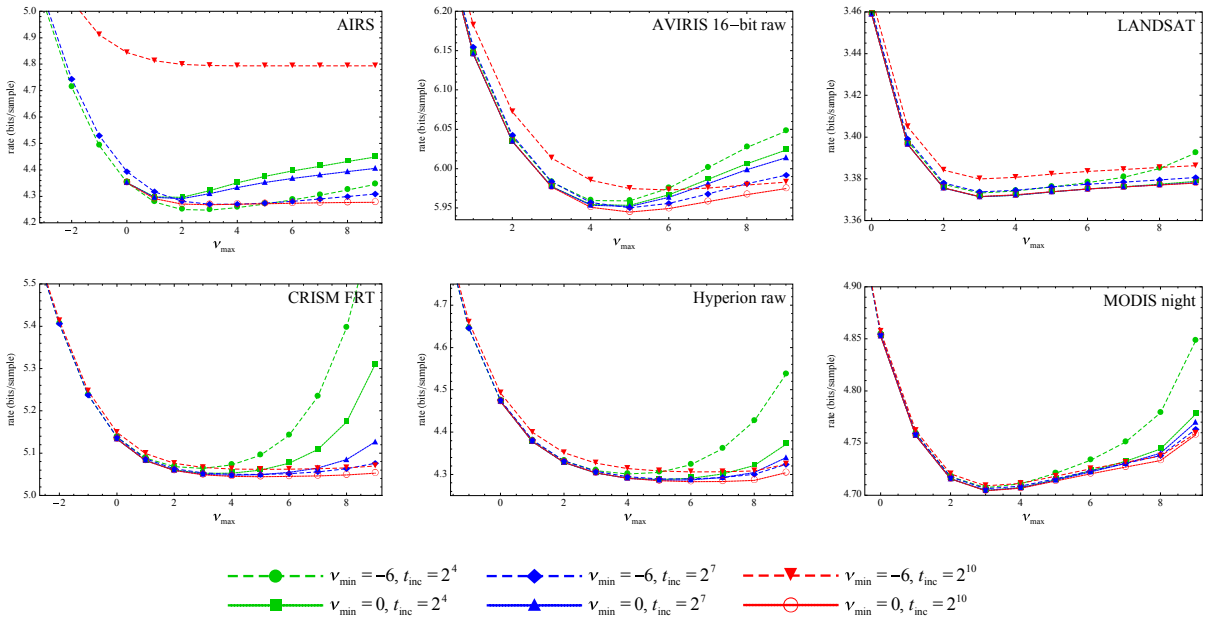


Figure 4-6: Average Compressed Data Rate for Different Choices of Parameters That Affect the Adaptation of the Predictor to the Image

For the images shown here, v_{max} is the most important weight adaptation parameter, and the values of t_{inc} and v_{min} have little impact except for the AIRS image. Given the small spatial size of this image (90×135), the combination of a large value of t_{inc} and a large difference $v_{max} - v_{min}$ ensures that the weight update scaling exponent does not reach its final value until near the end of the image.

4.2.4 WEIGHT RESOLUTION

The weight resolution parameter Ω determines the resolution with which weight values are represented. Thus increasing the value of Ω can provide more accurate prediction resulting in more effective compression in return for higher implementation complexity.

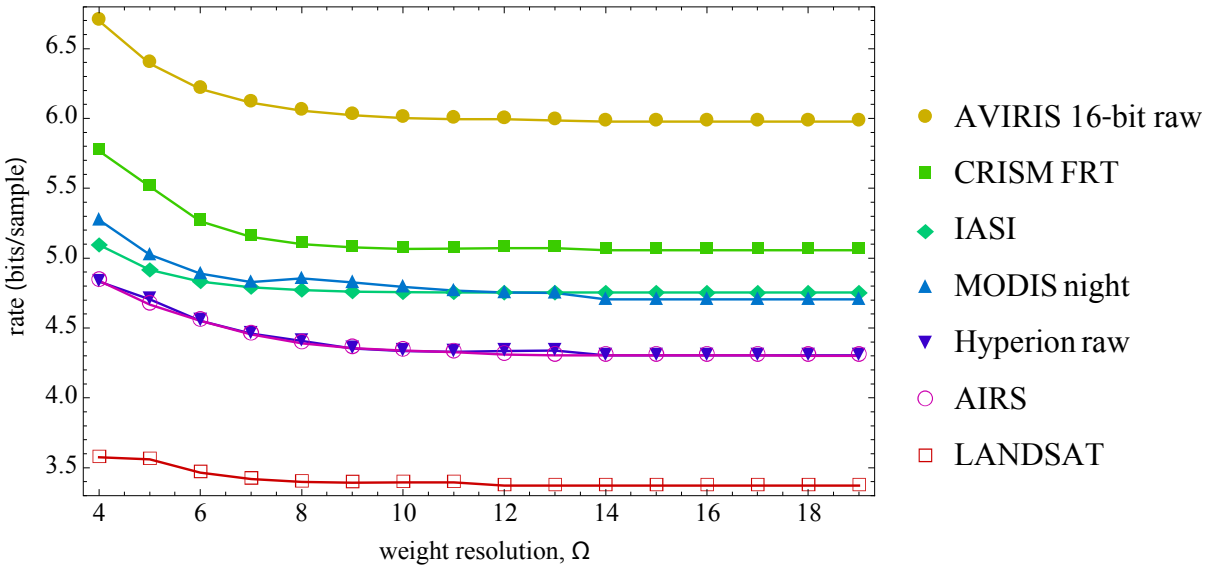


Figure 4-7: Average Bit Rate as a Function of Weight Resolution Ω Using v_{max}=3

Figure 4-7 shows the average compressed data rate as a function of weight resolution for several of the imaging instruments in the corpus. Using values of Ω near the minimum allowed can significantly hurt compression effectiveness. Beyond a certain point, however, increasing Ω may have little or no impact on compressed bit rate. In particular, in the weight update procedure, equation (34) of reference [1], the amount by which a weight value is incremented is

$$\left\lfloor \frac{1}{2}\left(u \cdot \mathrm{sgn}^{+}\left[e_z(t)\right] \cdot 2^{-\rho(t)} + 1\right) \right\rfloor = \left\lfloor u \cdot \mathrm{sgn}^{+}\left[e_z(t)\right] \cdot 2^{-\rho(t)-1} + \frac{1}{2} \right\rfloor \tag{19}$$

where u is some integer local difference value. Since $\mathrm{sgn}^{+}[e_z(t)] = \pm 1$, once the weight update scaling exponent $\rho(t)$ reaches a final (maximum) value of $v_{\max} + D - \Omega$, each weight increment is of the form

$$\left\lfloor \pm u \cdot 2^{\Omega - v_{\max} - D - 1} + \frac{1}{2} \right\rfloor. \tag{20}$$

If $\Omega \geq v_{\max} + D + 2$ then this weight increment is always a multiple of 2, and thus, in effect, the least significant bit of the weight value does not change with each update; i.e., the available weight resolution is not being fully used. Consequently, a reasonable rule-of-thumb is that using a weight resolution value larger than

$$\Omega^* = v_{\max} + D + 1 \tag{21}$$

is unlikely to improve compression effectiveness.

4.2.5 REGISTER SIZE

The main prediction calculation, equation (29) of reference [1], is defined to take into account the size of the register used to perform this calculation via the register size parameter R. If an implementation uses a sufficiently small register, then overflow may occur during prediction. When such an overflow occurs, the magnitude of the resulting prediction error is likely to be very large, and so compression effectiveness will suffer somewhat. Thus using a larger register in an implementation (that is, increasing the value of R) increases compression effectiveness at the expense of increased implementation complexity.

Even when an overflow occurs, compression will still be lossless because the decompressor performs the same prediction calculation, taking into account the register size R and thus duplicating any overflow that occurs in the compressor. Furthermore, both entropy coding approaches mitigate to some degree the extent to which poor prediction can lead to high bit rates.

One can bound the maximum value of register size R needed to ensure that overflow is mathematically impossible given the values of the weight resolution Ω, image bit depth D, number of bands for prediction P, and choice of full or reduced prediction mode.

For a given value of register size R, an overflow occurs when the quantity

$$\hat{d}_z(t) + 2^{\Omega}\left(\sigma_z(t) - 4s_{\text{mid}}\right) \tag{22}$$

cannot be represented as a signed R-bit quantity. Applying the bounds from table 3-1 in 3.2.5 to each term in the above expression, the range of possible values for this quantity is bounded by

$$\left[-2^{\Omega+1}\left(\left(2^D-1\right)(8P+\kappa)+1\right), 2^{\Omega+1}\left(\left(2^D-1\right)(8P+\kappa)-1\right)\right] \tag{23}$$

where the constant κ is

$$\kappa = \begin{cases} 1, & \text{reduced mode} \\ 19, & \text{full mode} \end{cases}. \tag{24}$$

A word size, in bits, sufficient to represent all values in this range is

$$R^* = \Omega + 2 + \left\lceil \log_2\left(\left(2^D-1\right)(8P+\kappa)+1\right)\right\rceil. \tag{25}$$

Thus if the register size R satisfies $R \geq R^*$, then overflow cannot occur in the prediction calculation. Table 4-1 tabulates the quantity R^* at maximum weight resolution, $\Omega = 19$. The table indicates that a 45-bit register is sufficient to guarantee that overflow will not occur in prediction for all possible choices of compression settings. The Recommended Standard requires $R \geq 32$ (see reference [1]), and so for some combinations of D, P, and Ω, overflow is impossible for any compliant implementation.

Table 4-1: Values of R*, the Register Size, in Bits, Sufficient to Ensure That Overflow Is Not Possible in the Prediction Calculation under Reduced Mode (Left) and Full Mode (Right) at the Maximum Value of Weight Resolution, Ω=19

	P=0	P=1	P=2	P=3	P=4	P=5	P=6	P=7	P=8	P=9	P=10	P=11	P=12	P=13	P=14	P=15
D=2	23	26	27	28	28	28	29	29	29	29	29	30	30	30	30	30
D=3	24	27	28	29	29	30	30	30	30	30	31	31	31	31	31	31
D=4	25	29	29	30	30	31	31	31	31	32	32	32	32	32	32	32
D=5	26	30	31	31	31	32	32	32	32	33	33	33	33	33	33	33
D=6	27	31	32	32	33	33	33	33	33	34	34	34	34	34	34	34
D=7	28	32	33	33	34	34	34	34	35	35	35	35	35	35	35	35
D=8	29	33	34	34	35	35	35	35	36	36	36	36	36	36	36	36
D=9	30	34	35	35	36	36	36	36	37	37	37	37	37	37	37	37
D=10	31	35	36	36	37	37	37	37	38	38	38	38	38	38	38	38
D=11	32	36	37	37	38	38	38	38	39	39	39	39	39	39	39	39
D=12	33	37	38	38	39	39	39	39	40	40	40	40	40	40	40	40
D=13	34	38	39	39	40	40	40	40	41	41	41	41	41	41	41	41
D=14	35	39	40	40	41	41	41	41	42	42	42	42	42	42	42	42
D=15	36	40	41	41	42	42	42	42	43	43	43	43	43	43	43	43
D=16	37	41	42	42	43	43	43	43	44	44	44	44	44	44	44	44

	P=0	P=1	P=2	P=3	P=4	P=5	P=6	P=7	P=8	P=9	P=10	P=11	P=12	P=13	P=14	P=15
D=2	27	28	28	29	29	29	29	29	29	30	30	30	30	30	30	30
D=3	29	29	29	30	30	30	30	31	31	31	31	31	31	31	31	31
D=4	30	30	31	31	31	31	31	32	32	32	32	32	32	32	32	33
D=5	31	31	32	32	32	32	33	33	33	33	33	33	33	33	33	34
D=6	32	32	33	33	33	33	34	34	34	34	34	34	34	34	35	35
D=7	33	33	34	34	34	34	35	35	35	35	35	35	35	35	36	36
D=8	34	34	35	35	35	35	36	36	36	36	36	36	36	36	37	37
D=9	35	35	36	36	36	36	37	37	37	37	37	37	37	37	38	38
D=10	36	36	37	37	37	37	38	38	38	38	38	38	38	38	39	39
D=11	37	37	38	38	38	38	39	39	39	39	39	39	39	39	40	40
D=12	38	38	39	39	39	39	40	40	40	40	40	40	40	40	41	41
D=13	39	39	40	40	40	40	41	41	41	41	41	41	41	41	42	42
D=14	40	40	41	41	41	41	42	42	42	42	42	42	42	42	43	43
D=15	41	41	42	42	42	42	43	43	43	43	43	43	43	43	44	44
D=16	42	42	43	43	43	43	44	44	44	44	44	44	44	44	45	45

Even when overflow is mathematically possible, it may be very rare for real images. For the majority of the images in the corpus, the increase in compressed data rate due to register overflows is less than one percent (using P=3 and all other default settings) even when the register size parameter R is set to the minimum allowed value. But for some images, the increased bit rate due to overflows can be significant when the register size is sufficiently small. Figure 4-8 shows compressed bit rate as a function of register size when R varies from the minimum value allowed by the standard, $\max\{32, D + \Omega + 2\}$, up to R^*.

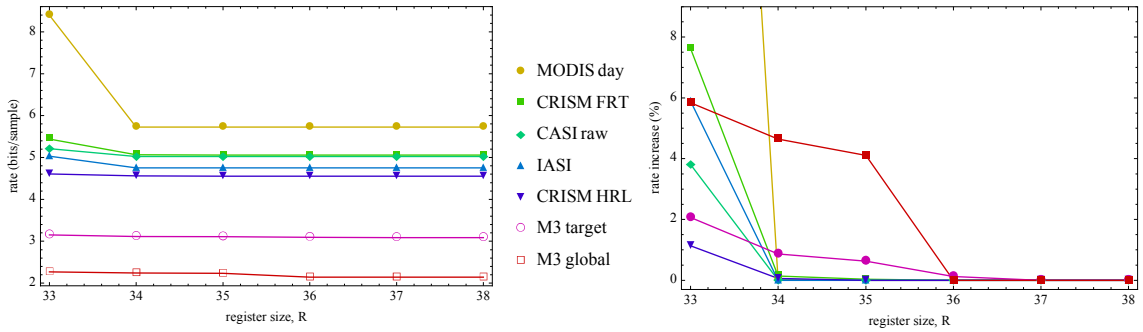


Figure 4-8: Compressed Bit Rate (Left) and Percentage Increase in Rate (Right) as a Function of Register Size *R*, Averaged over All Images of a Given Type

4.2.6 CUSTOM WEIGHT INITIALIZATION

The custom weight initialization option allows a user to select initial weight vectors that might provide more effective compression than the default values for a given input image. When this option is used, the user selects the value of the weight initialization resolution, Q, and the user specifies the first Q bits of each initial weight value via the weight initialization table, $\{\Lambda_z\}_{z=1}^{N_z}$. The weight initialization table may be encoded in the header, or may be omitted when it is known in advance to the decompressor, which might arise, e.g., if the same initial weight vectors are used throughout a mission phase.

Custom weight initialization may provide improved compression performance, for example, when training data are available that would allow the development of a set of initial weight vectors specifically tuned to the imaging instrument. Alternatively, custom weights might be selected based on a recently compressed image. For example, for a pushbroom or whiskbroom hyperspectral imager one might naturally partition the imager output along the y-axis to split a large image into several smaller images that can be independently decompressed, thus providing some robustness to loss of compressed data on the communications channel. In this case, one could set the set of initial weight vectors for an image to be equal to quantized versions of the final weight vectors for the preceding image.

An experiment was performed to illustrate the improvement provided by employing a custom weight initialization. In the experiment there are two images: a training image and a test image. Both images are in fact continuous pieces of a larger image, with the training image immediately preceding the test image along the y-axis, and both entirely covering the larger image along the x- and z-axes (see figure 4-9).

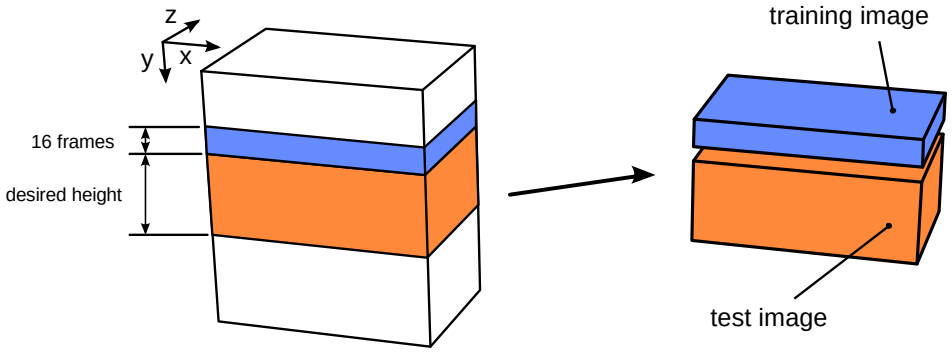


Figure 4-9: Training and Test Images Employed to Assess the Effects of Custom Weight Initialization

The experiment is as follows. First, the training image is compressed without using custom weights. The resulting compressed training image is discarded, but the values of the weight vectors at the end of the compression process are preserved. Then, the test image is compressed twice; once without using custom weights, and once using custom weights obtained from the aforementioned preserved weight vectors of the previous compression. This allows measurement of the effects of both weight initialization options. In the

experiment, to cover different values of the weight vectors, these two steps are repeated for multiple training and test images, which are extracted at multiple locations of a larger image. Averaged results are reported.

This experiment is repeated multiple times for varying test image heights, to account for the diminishing impact of custom weight initialization as the height of the test image increases (in all cases the height of the training image is 16). Similarly, it is also repeated multiple times to assess the impact of the custom weight vectors resolution.

In all cases $v_{\max} = 3$ and $t_{\text{inc}} = 2^6$ are used as initial settings, and the values of $v_{\min}$ and Ω are varied when custom weights are used, but kept fixed at $v_{\min} = -1$ and $\Omega = \min\{v_{\max} + D + 1, 19\}$ when default weights are used. The extra overhead needed to encode custom weights (via the weight initialization table) is included in bit rate calculations.

Figure 4-10 shows the improvement obtained by using custom weight initialization as a function of image height in this experiment when $v_{\min} = 3$ and the weight initialization resolution is fixed at $Q = \lfloor \Omega / 2 \rfloor$. Multiple heights are simulated for the bottom image by compressing only as many frames of the image as needed to reach the required height, and ignoring the remaining ones (those on the bottom of the image). Figure 4-11 shows the average improvement due to custom weights as a function of Q when $v_{\min} = 3$ at image heights of 16 and 32. Figure 4-12 shows the average improvement as a function of $v_{\min}$ when $Q = \lfloor \Omega / 2 \rfloor$ at image heights of 16 and 32.

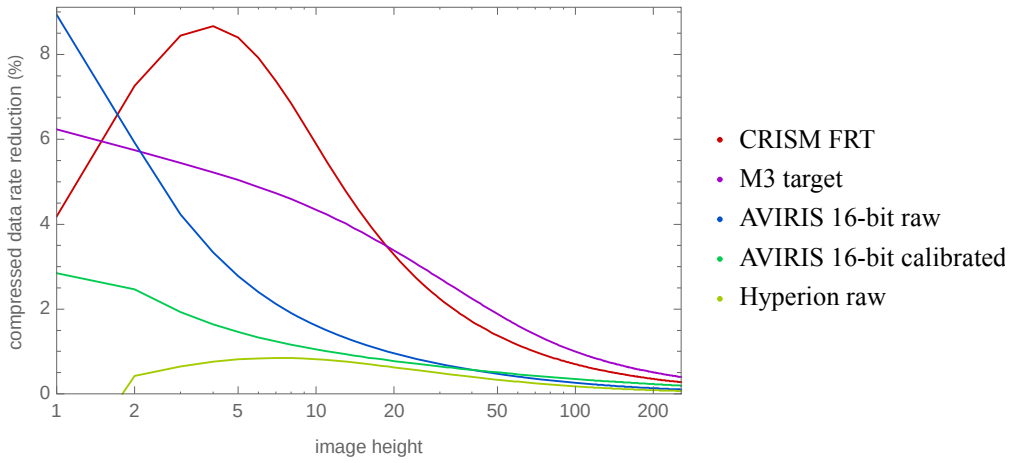


Figure 4-10: Average Reduction in Compressed Data Rate Obtained by Using Custom Weight Initialization (with $v_{\min} = 3$, $Q = \lfloor \Omega / 2 \rfloor$), Compared to Default Weight Initialization, as a Function of Image Height

NOTE – Larger values are better.

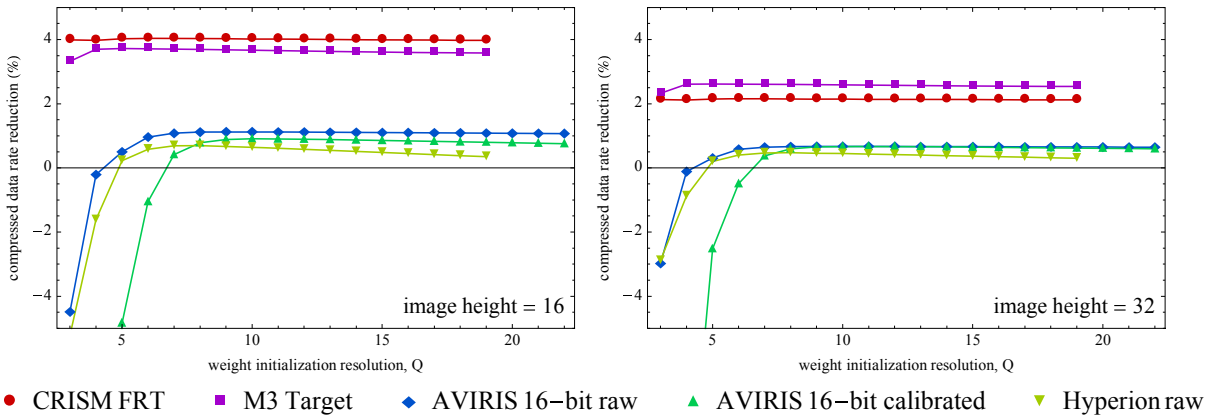


Figure 4-11: Average Reduction in Compressed Data Rate Obtained by Using Custom Weight Initialization (with $v_{min} = 3$, for Image Heights 16 and 32), Compared to Default Weight Initialization, as a Function of Weight Initialization Resolution, Q

NOTE – Larger values are better.

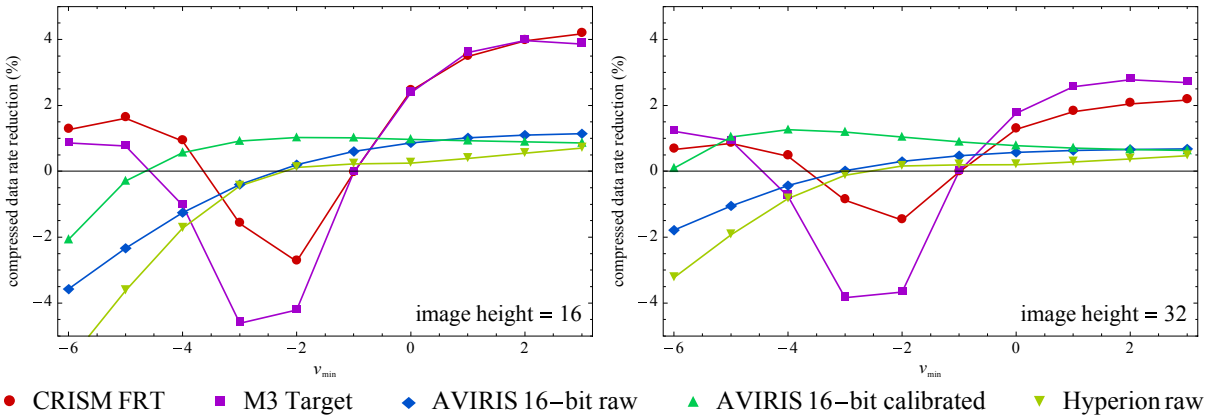


Figure 4-12: Average Reduction in Compressed Data Rate Obtained by Using Custom Weight Initialization (with $Q = \lfloor \Omega / 2 \rfloor$, for Image Heights 16 and 32), Compared to Default Weight Initialization, as a Function of v_{min}

NOTE – Larger values are better.

The results indicate that when initial weight vectors are well chosen, the use of custom weight initialization can provide a modest improvement in compression effectiveness, especially for small images. As would be expected, this benefit diminishes for larger images because the weights continue to adapt to the image data during compression. For very small images (such those consisting of less than five frames), the cost of signaling a custom weight initialization may surpass the improved performance it provides, as the additional header information due to custom weights is averaged over a small number of pixels. The results of

this experiment are relatively insensitive to the choice of weight initialization resolution, Q, provided that the smallest allowed values are avoided. Larger values of $v_{\min}$ may be necessary to realize the benefit of well-chosen custom weight vectors.

The improved effectiveness for small images provided by the use of custom weight initialization does not imply that in general a higher compression performance is obtained for small images in relation to large images; in fact, the opposite should be expected.

4.3 ENTROPY CODER

4.3.1 INTRODUCTION

Subsections 4.3.2 and 4.3.3 discuss some of the key parameters associated with the sample-adaptive and block-adaptive encoding approaches respectively. Subsection 4.3.4 discusses some of the differences between these two encoders.

4.3.2 SAMPLE-ADAPTIVE ENTROPY CODER SETTINGS

4.3.2.1 Initialization and Adaptivity

The sample-adaptive entropy coder state is initialized via the initial count exponent γ_0, which controls the initial counter value, and the accumulator initialization table $\{k'_z\}_{z=0}^{N_Z-1}$, which controls the initial value of the accumulator Σ_z in each band z. For simplicity, the discussion here considers only the case where an accumulator initialization constant K is used, i.e., $k'_z = K$ for all z.

The rescaling counter size parameter, γ^*, controls the interval with which the counter and accumulator are rescaled. A smaller value of γ^* yields faster adaptation to changing source statistics, but potentially worse steady-state performance. Figure 4-13 shows the relative change in compressed bit rate as a function of γ^* for several different combinations of accumulator initialization constant K and initial count exponent γ_0. The figure suggests that performance is generally improved by avoiding the largest allowed values of K. Using a smaller value of K not only tends to improve compression effectiveness, but also seems generally to make compression performance fairly insensitive to the choice of γ_0 and γ^*.

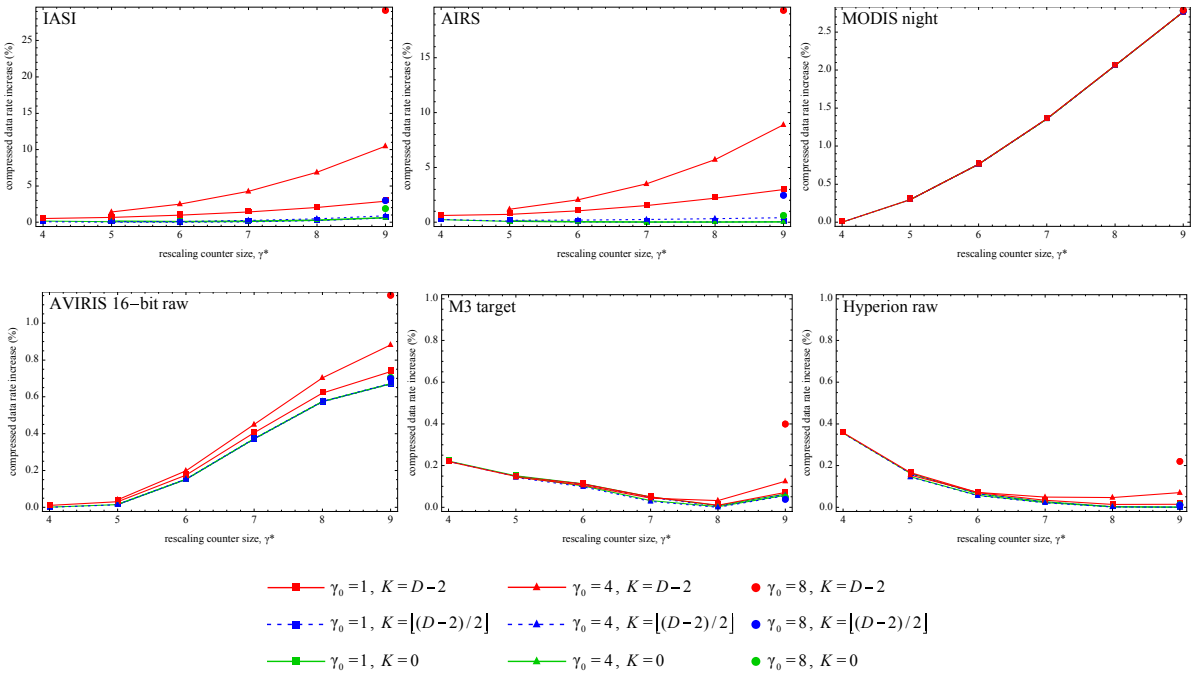


Figure 4-13: Change in Compressed Data Rate (I.e., Relative Increase Compared to the Data Rate Obtained by the Optimum Combination of γ_0, γ^*, and K) as a Function of Rescaling Counter Size γ^*, for Different Combinations of Accumulator Initialization Constant K and Initial Count Exponent γ_0

NOTE – Smaller is better.

It is observed in figure 4-13 that the values of γ_0, γ^*, and K have a more dramatic impact on compression effectiveness of IASI and AIRS images than the others. Because the initial count exponent γ_0 and the accumulator initialization constant K affect only the initialization of entropy coder state variables for each spectral band, their effect on compressed bit rate quickly diminishes for an input image whose spatial size (i.e., the product $N_X N_Y$) is large; IASI and AIRS images have small spatial size and so the values used for γ_0 and K have more of an impact on compression effectiveness than for other images in the corpus. To demonstrate that this effect is due to spatial size, rather than some other characteristic of the image data, figure 4-14 shows the relative change in compressed bit rate from reducing the spatial size of AVIRIS 16-bit raw, M3 target, and Hyperion raw images by partitioning each image along the y axis into smaller images, each with height $N_Y = 16$, except possibly the last image. For these smaller images, compression effectiveness becomes more sensitive to the choice of K, and to a lesser degree, the choice of γ_0. Also, on smaller images, the use of a smaller value of γ^* causes the impact of the values of γ_0 and K to diminish more rapidly and offer improved performance, particularly when a poor choice of K is used.

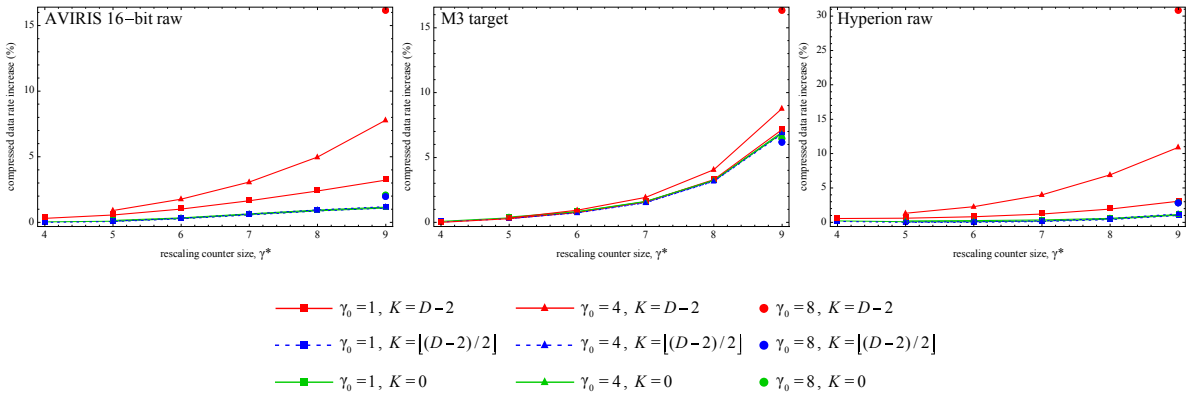


Figure 4-14: Change in Compressed Data Rate as a Function of Rescaling Counter Size γ^*, for Different Combinations of Accumulator Initialization Constant *K* and Initial Count Exponent γ_0, When an Image Is Partitioned into Smaller 16-Frame Images That Are Each Independently Compressed

4.3.2.2 Unary Length Limit

As described in 3.3.4, the sample-adaptive encoder uses an adaptively updated estimate of the mean value of the mapped prediction residual δ in the current spectral band to select from a family of variable-length binary codes to encode each sample. The codes used are length-limited GPO2 codes.

The family of codes is indexed by integer parameter k (written as $k_z(t)$ in the Recommended Standard to indicate that the value of the parameter varies with the sample being encoded). The code used to encode a sample, i.e., the value of k, is selected based on an estimate of the mean value of the mapped prediction residual in the current spectral band, as described in 3.3.4. Smaller mapped prediction residual values (i.e., more accurate predictions) are encoded using shorter codewords. The value of k controls this tradeoff; smaller values of k correspond to increasing confidence in prediction accuracy, using fewer bits to encode smaller values of δ, in return for a higher cost to encode larger values of δ. Figure 4-15 illustrates the codeword lengths of the family of codes available when the input image has dynamic range D=12 bits and the unary length limit is $U_{\max} = 20$.

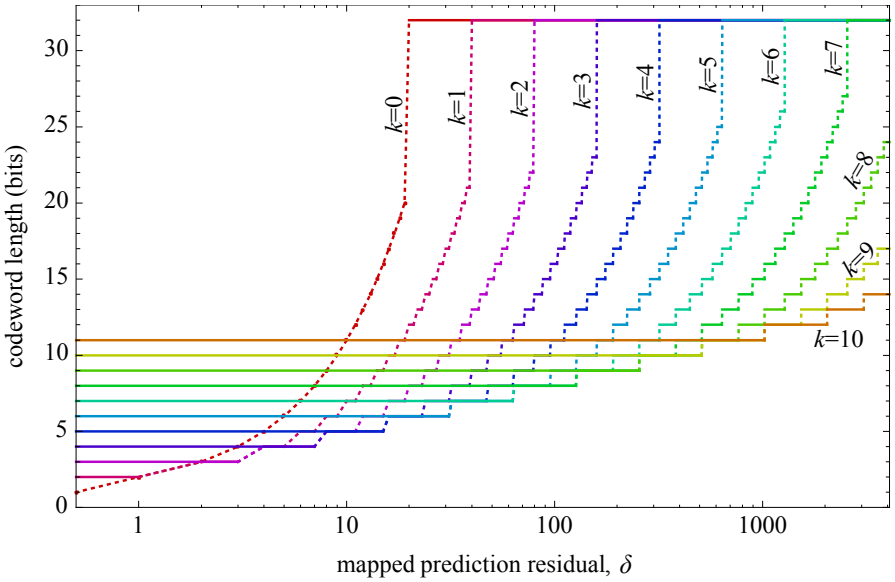


Figure 4-15: Codeword Lengths for the Set of Length-Limited GPO2 Codes Used by the Sample-Adaptive Entropy Coder When Image Dynamic Range Is D=12 Bits and the Unary Length Limit Is U_{max} = 20

The maximum codeword length is $U_{max} + D$ bits. Decreasing the value of the unary length limit reduces the cost to encode poorly predicted samples (i.e., larger values of mapped prediction residual), in return for less efficient coding of some smaller values of δ, as can be observed in figure 4-16, which illustrates the codeword length function for the k=3 code, at minimum and maximum values of U_{max}, when the input image has dynamic range D=12 bits.

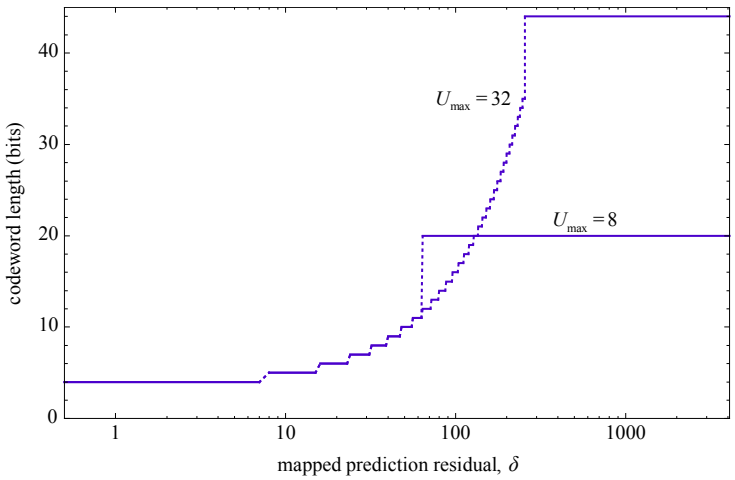


Figure 4-16: Codeword Lengths for k=3 Length-Limited GPO2 Codes Used by the Sample-Adaptive Entropy Coder at Minimum and Maximum Allowed Values of the Unary Length Limit U_{max} When Image Dynamic Range Is D=12 Bits

Because the sample-adaptive encoder adaptively selects the code (i.e., the value of k) based on the prediction accuracy of recently encoded samples in the spectral band, for many images the codeword length limit is reached for only a small fraction of samples, and thus coding performance is not very sensitive to the value of $U_{\max}$. This is evidently true for images in the corpus; figure 4-17 shows that the increase in bit rate produced by using a suboptimal value of $U_{\max}$ is quite small even when the minimum allowed register size is used (which increases the likelihood of register overflows, and thus outlier samples).

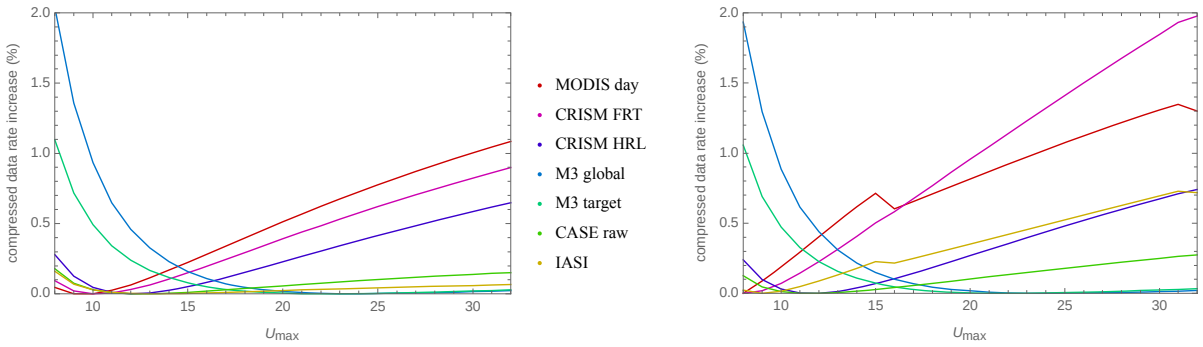


Figure 4-17: Impact of Changing the Unary Length Limit on Compressed Data Rate Shown for Maximum and Minimum Allowed Register Size R (Left and Right Respectively)

4.3.3 BLOCK-ADAPTIVE ENTROPY CODER SETTINGS

4.3.3.1 Overview

The block-adaptive encoding method partitions the mapped prediction residuals into blocks and selects the coding method for each block based on the block's contents. When this coding method is used, the sample encoding order and block size are the entropy coding settings that have the biggest impact on compressed data rate.

4.3.3.2 Sample Encoding Order

Under the block-adaptive encoding approach, the set of mapped residual values that make up a block depends on the sample encoding order. For example, under BSQ encoding order, most blocks will be made up of samples from the same band, while under BIP order, samples in a block will generally come from several different bands. Consequently, under the block-adaptive entropy coding option, the method used to encode a given sample depends on the other samples in the block, which in turn depends on the encoding order. Thus compressed data rate depends on the sample encoding order when the block-adaptive encoder is used, but not when the sample-adaptive encoder is used.

Figure 4-18 shows the compressed bit rates achieved for different types of images in the corpus when BSQ, BIL, and BIP encoding orders are used with block-adaptive coding, along with results for sample-adaptive coding.

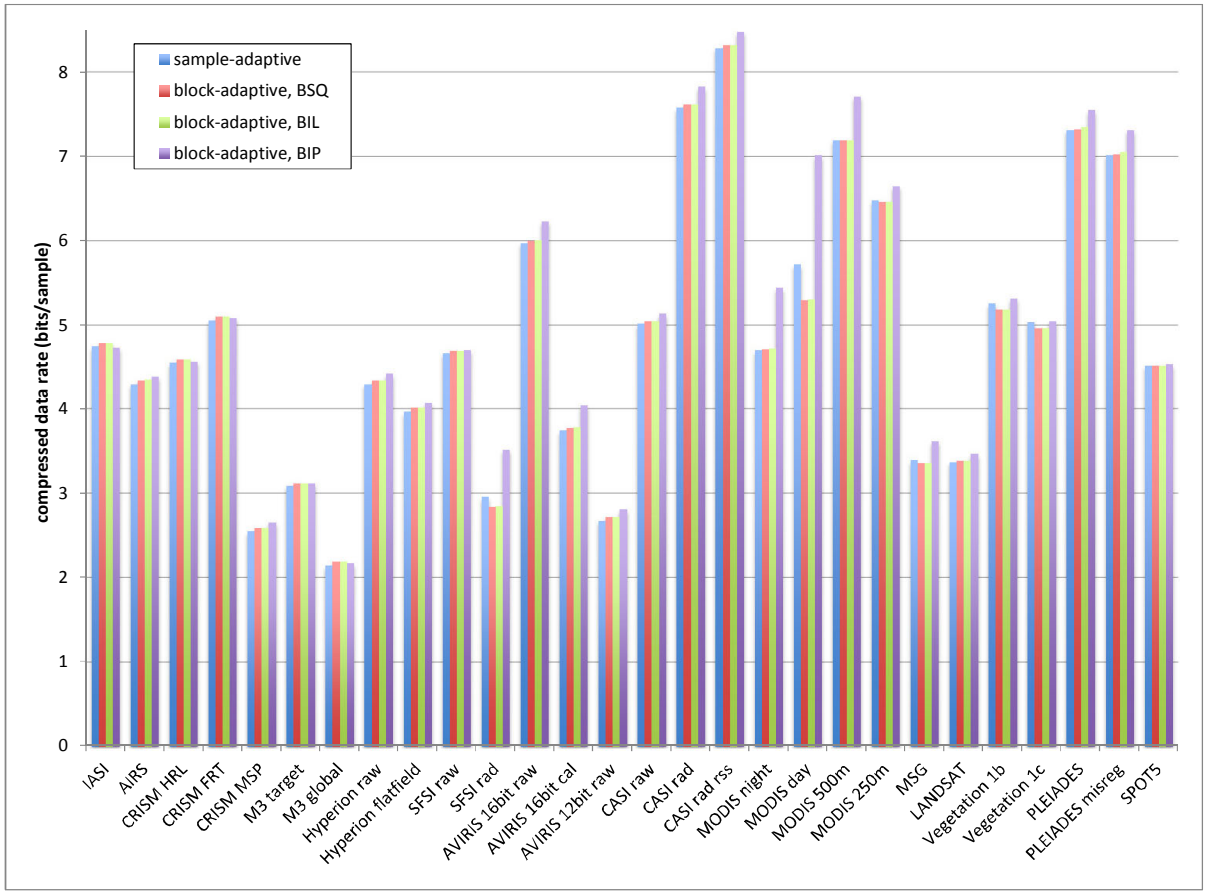


Figure 4-18: Average Compressed Data Rate for Images in the Corpus When the Sample-Adaptive Entropy Coder Is Used, and When the Block-Adaptive Encoder Is Used Under BSQ, BIL, and BIP Sample Orders for Block Size *J*=64

A few trends are observed when block-adaptive coding is used:

– BIP encoding order typically does not perform as well as BIL or BSQ. In cases where BIP performs better, it is generally not by much. (BIL and BSQ tend to produce blocks of samples that are all from the same band, whereas BIP mixes samples from different bands in each block, and so this result is to be expected.)

– BIL and BSQ generally give nearly the same performance. When there is a difference, BSQ is generally better, but only by a small amount.

When a block of mapped prediction residuals is all zeros (which arises when there is no prediction error for any sample in the block), the block-adaptive encoder's 'zero block'

coding method is used (see subsection 3.4.3 of reference [13]). The data rate to encode such an all-zeros block depends on the number of consecutive all-zeros blocks, but is always less than one bit per sample. By contrast, the sample-adaptive encoder always uses at least one bit to encode each sample.

Sample-adaptive coding usually gives a small compression advantage over block-adaptive coding regardless of the sample encoding order. But there are counterexamples where block-adaptive coding (under BSQ or BIL order) outperforms sample-adaptive coding, and this is apparently due to the high compression effectiveness of the zero block coding.

- In the SFSI rad image, 20 of the 240 bands are all zeros. About 8 percent of the blocks of mapped predictions residuals are all zeros, and can thus take advantage of the zero-block coding options available under the block-adaptive coding method. The net result is about 4 percent lower compressed bit rate than sample-adaptive coding.
- In the MODIS day images, in the first 11 (of 14 total) bands, nearly 60 percent of the samples are saturated, i.e., bands have large regions of constant-valued samples, leading to blocks of all-zeros mapped prediction residuals that are encoded at very low bit rate under the block-adaptive coding method.

When the image width is a multiple of the block size J, both BIL and BSQ encoding orders produce the same set of blocks. However, the order of these blocks is not the same, and so the compressed bit rate may still be different because all-zero blocks may be more likely to be adjacent under BSQ order than BIL order.

4.3.3.3 Block Size

Under the block-adaptive coding approach, mapped prediction residuals are partitioned into blocks of J samples, and a coding method is independently selected for each such block. A few overhead bits are used to indicate the coding method selected for each block, and the bit rate cost of this overhead increases with smaller values of J. Specifically, for the allowed block size values of 8, 16, 32, and 64, the bit rate due to overhead is 0.5, 0.25, 0.125, and 0.0625 bits/sample, respectively, for blocks that are not all zeros when $D \geq 9$.

In practice, using a smaller block size generally results in higher compressed data rate; however, the increase is less than the bound obtained from the difference in bit rate due to overhead, because the use of a smaller value of J allows the entropy coder to adapt more rapidly to changing source statistics. In fact, one can see from figure 4-19 that the largest value of block size is not always optimum. In particular, for the MODIS day images, the use of the smallest block size provides the most effective compression, presumably because it increases the fraction of blocks that are all zeros and are thus economically encoded using the zero-block coding method.

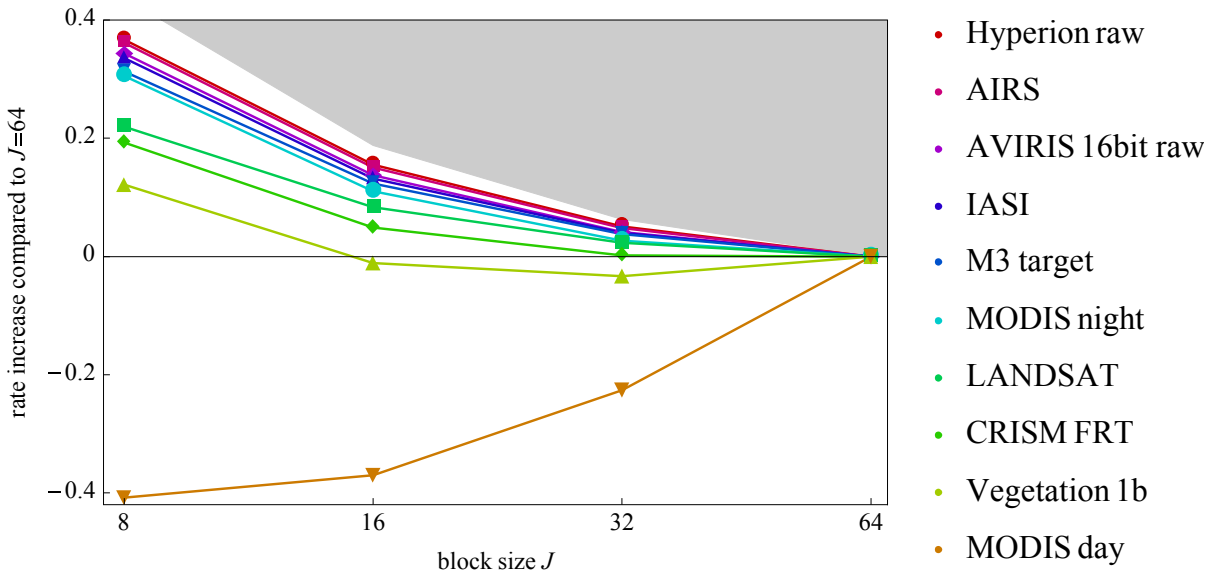


Figure 4-19: Average Increase in Compressed Bit Rate, Compared to the Use of Block Size J=64, under Block-Adaptive Coding Using BSQ Sample Order

NOTE – The gray region shows the upper bound on this increase derived from the cost of overhead bits for blocks that are not encoded using the zero-block option.

4.3.4 DIFFERENCES BETWEEN ENTROPY CODING OPTIONS

Comparing compression effectiveness of the two encoders, based on results above using images in the corpus, it is noted that:

- Block-adaptive coding provides more effective compression of long sequences of perfectly predicted samples. In practice, this tends to arise for image bands having large contiguous regions of saturated or all-zeros samples, and when BIL or BSQ encoding order is used so that these samples tend to be adjacent in the encoding order. In this case, the benefit of block-adaptive coding can be substantial.
- Under BIP encoding order, sample-adaptive encoding nearly always provides more effective compression, sometimes by a substantial amount.
- Under BIL and BSQ encoding orders, sample-adaptive encoding typically provides slightly more effective compression than block-adaptive coding for hyperspectral images. For multispectral images, results are more mixed but generally favor block-adaptive encoding.

Differences in implementation between the two entropy coding options are discussed in 5.4.3.4.

5 IMPLEMENTATION ISSUES

5.1 INTRODUCTION

This section discusses some practical issues that may be of interest to implementers of the Recommended Standard. Subsection 5.2 notes a useful property of the compressor that makes it easy for a compressor that handles unsigned input images to be used for signed input images, and vice versa. Subsection 5.3 discusses the impact of bit errors and data loss on the reconstructed image and illustrates how partitioning an image into smaller independently decompressible images can help to limit the impact of these events. Subsection 5.4 discusses hardware implementation of the Recommended Standard. Subsection 5.5 briefly examines the impact on compression effectiveness of nonuniformities in the detector and considers the possibility of improving compression performance by pre-processing the image to reduce artifacts arising from these nonuniformities.

5.2 SIGNED AND UNSIGNED IMAGES

The compressor has the following property:

> *A denotes an unsigned input image with bit depth D. A′ denotes a signed image produced by subtracting* $s_{\text{mid}} = 2^{D-1}$ *from every sample of A. Given a fixed set of compression settings, the compressed bitstreams for A and A′ are identical, except for the one-bit header field 'Sample Type' in the Image Metadata portion of the header, which indicates whether the image is signed or unsigned.*

This property can simplify implementation, because a compressor that supports only signed images can easily be extended to also support unsigned images, and vice-versa.

5.3 DEALING WITH DATA LOSS ON SPACE COMMUNICATIONS CHANNELS

5.3.1 THE IMPACT OF BIT ERRORS AND DATA LOSS

Data transmitted over space communications channels are vulnerable to corruption in the form of data loss and/or bit errors. While such events may occur with low probability, even a single bit error in a compressed image generally results in corruption of reconstructed samples extending to the end of the image, because reconstruction of future samples depends on accurate reconstruction of past samples. As an example, figure 5-1 shows a false-color image derived from reconstructed bands of a hyperspectral image following a single bit error occurring near the midpoint of the compressed image data.

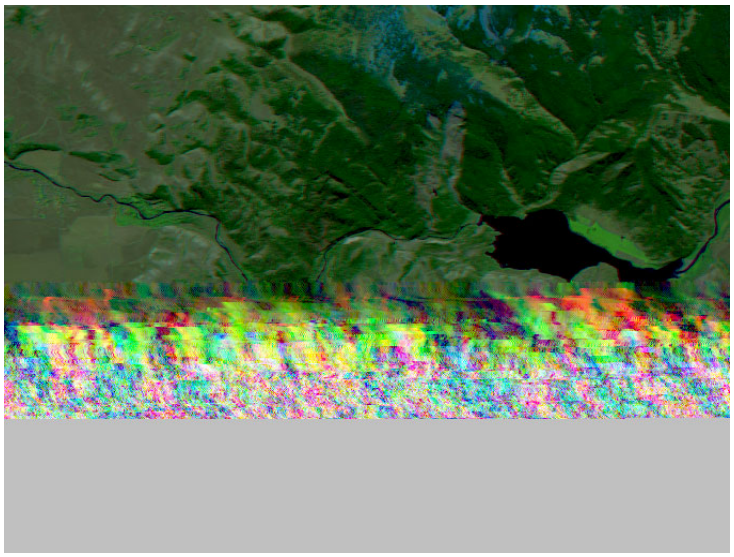


Figure 5-1: Example of the Impact of a Bit Error in the Compressed Image

NOTE – This false-color image was derived from three bands of a reconstructed image following a single bit error in the compressed image, which was encoded in BIP order.

The fact that reconstructed image data are corrupted is obvious to a human observer of figure 5-1, and this is nearly always the case unless the bit error occurs very close to the end of the image. Moreover, it is straightforward to produce a decompressor that will (with very high probability) automatically detect that a bit error has occurred. The decompressor simply needs to check that it has reconstructed $N_X \cdot N_Y \cdot N_Z$ image samples and exhausted the compressed image data at the same time. In the example of figure 5-1, the decompressor exhausted the compressed image data before reconstructing enough samples to complete the image; the missing sample values are shown in gray at the bottom of the image.

All of the intact compressed data preceding a data loss or error event can be used to recover some portion of the image; the sample encoding order affects what data are recovered. In the example of figure 5-1, samples were encoded in BIP order and frames preceding the bit error (i.e., the upper spatial portion of the image) were recovered. If BSQ encoding order had been used, then all samples in some number of initial spectral bands would have been recovered while later spectral bands would have been corrupted or missing.

To protect against the dramatic impact that a bit error can have on reconstructed imagery, the systems engineer should set an appropriate error rate requirement for the communications link. If a mission requires compressed images to be recovered with some given probability, a corresponding error rate requirement can be derived for the transmission of *packets* (reference [4]), *encapsulation packets* (reference [5]), *files* (reference [6]), or *transfer frames* (references [7] and [19]). This may lead to the selection of appropriate channel codes (references [20], [21], [43], and [44]) capable of offering the required protection of these data structures. (If uncoded transmission is desired, the image recovery probability requirement may be used to derive a bit error rate requirement for the channel.)

5.3.2 PARTITIONING AN IMAGE INTO SMALLER SEGMENTS

To limit the effects of data corruption on reconstructed imagery, one can partition a large image into smaller images that can be independently decompressed, as illustrated in figure 5-2. In this discussion, each of these smaller images is referred to as a *segment* of the larger image. It is assumed that communications protocols employed by the spacecraft incorporate mechanisms (e.g., a packetization structure) that allow the beginning of the next image segment to be identified following a data corruption event, and thus the impact of data corruption is limited to the affected image segment. The Recommended Standard does not directly address such image segmentation; the term 'segment' is not part of the standard. In the view of the Recommended Standard, each such image segment is simply a separate image.

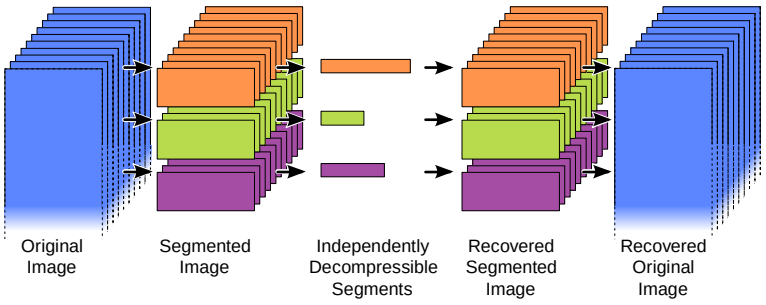


Figure 5-2: Overview of Image Segmentation

In figure 5-2 and in the example below, image segments are produced by partitioning a larger image along the *y*-axis. This is a natural approach for imagers that produce data in BIP or BIL order, but other partitioning approaches could also be used.

Using smaller segments provides increased robustness to data corruption, but reduces compression effectiveness because

a) each compressed segment includes the overhead cost of an image header;

b) samples at segment boundaries have fewer neighbors for use in prediction; and

c) the predictor and entropy coder take some time to adapt, and so samples near the beginning of a segment will, on average, not be compressed as effectively as later samples.

To mitigate this compression performance reduction due to segmentation, the Recommended Standard allows information about the state of the encoder at the end of one segment to be optionally included in the header of the next segment to control the initialization of compressor state variables, specifically via custom weight initialization (see 4.2.6) and the accumulator initialization table. Thus the impact of c) is reduced at the expense of a slight increase in a).

As an example illustrating that compression effectiveness tends to increase by using larger segments and by using custom weight initialization, compressed data rate is measured for two of the test images after partitioning into smaller image segments of a given height, and

using weight resolution $\Omega = \min\{D+4,19\}$. For the first segment, default weight initialization is used and $v_{\min} = -1$. For all subsequent segments custom weight initialization is used with $v_{\min} = 3$, $Q = \lfloor \Omega / 2 \rfloor$, and with initial weights set to equal quantized versions of the final weights obtained from the preceding segment. This approach is compared to the use of the default settings indicated in annex C applied to each segment. Figure 5-3 shows the compressed data rate obtained for these two approaches. It is observed from the figure that the impact of segment height on compression effectiveness varies depending on the image, and that the use of custom weight initialization tends to improve compression performance when small image segments are used.

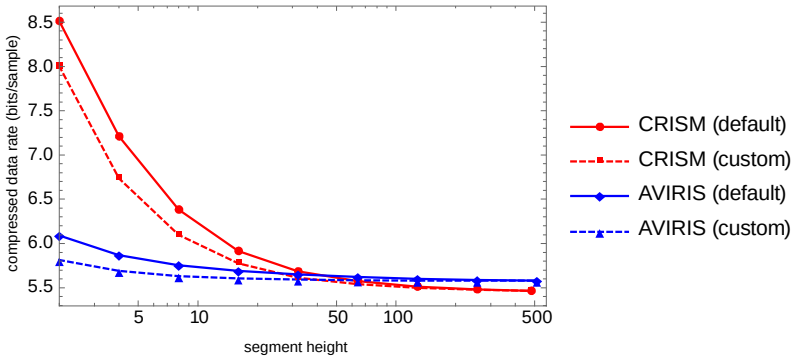


Figure 5-3: Compressed Data Rate as a Function of Segment Height for Test Images 'crism_frt00013e49_07_sc166' and 'aviris_sc10_raw'

Users should be judicious in partitioning an image into smaller and smaller portions (leading to decreasing compression effectiveness), particularly when data loss or corruption events are very rare.

5.4 HARDWARE IMPLEMENTATION

5.4.1 INTRODUCTION

The Recommended Standard describes the compression operations in a way that facilitates relatively low complexity hardware implementations on FPGAs or ASICs. The algorithm works with only integer arithmetic, and all mathematical operations can be implemented with fixed shifters, barrel shifters, multipliers, and adders. No divisions are necessary.

5.4.2 BACKGROUND

Aranki *et al.* have implemented the FL compressor on which the present Recommended Standard is based in Xilinx Virtex-4 (references [22] and [23]) and Virtex-5 FPGAs (reference [24]). The original FL algorithm uses an adaptive coding approach using GPO2 codes, which has been formalized in the Recommended Standard as the *sample-adaptive* entropy coder option. The compression IP compresses samples in BIP order and runs at a clock speed of 40 MHz. One sample is compressed each clock cycle, resulting in a

throughput of 40 Msample/sec. The implementation has a rather low device utilization of the Xilinx Virtex-5 XC5VSX50T (38 percent of Slice LUTs).

A low-complexity implementation of the Recommended Standard was implemented by Santos *et al.* (reference [25]) with the sample-adaptive encoding option only. Synthesis results were obtained on a space-qualified RTAX1000S and a Virtex-5 XC5VFX130. The compression IP has an occupancy of 34 percent of the RTAX1000S with a maximum frequency of 43 MHz, resulting in a throughput of 4 MSamples/sec. When synthesized in a Virtex-5 XC5VFX130, it results in very low device utilization (2 percent of Slice LUTs). The maximum frequency of this implementation is 134 MHz, which yields a throughput of 11.30 MSamples/sec for a configuration with P = 3 and BSQ compression order.

5.4.3 MAIN CONSIDERATIONS

5.4.3.1 General

When developing a hardware implementation of the Recommended Standard it is necessary to take account of the factors that contribute the most to its complexity. In general, the complexity of the mathematical operations in the Recommended Standard is low; therefore the following discussion focuses on identifying the main constraints in terms of storage requirements and throughput limitations caused by data dependencies.

General notes are provided here about complexity of prospective hardware implementations of the Recommended Standard; the actual storage requirements and throughput limitations will depend on the user's designed compression architecture and target technology. An example of a simplified schematic of a possible implementation using sample-adaptive coding is shown in figure 5-4, in which the blocks represent the main stages of the Recommended Standard.

5.4.3.2 Local Sum and Local Differences Calculation

The predictor in the Recommended Standard is causal; i.e., only previously processed neighboring samples are needed to compute the local sum and local differences. In order to avoid the impact on latency of having to read the necessary neighboring samples, it is possible to adequately arrange the previously processed samples temporarily in a storage component, such as a RAM memory or a FIFO, as shown in (references [22] and [25]).

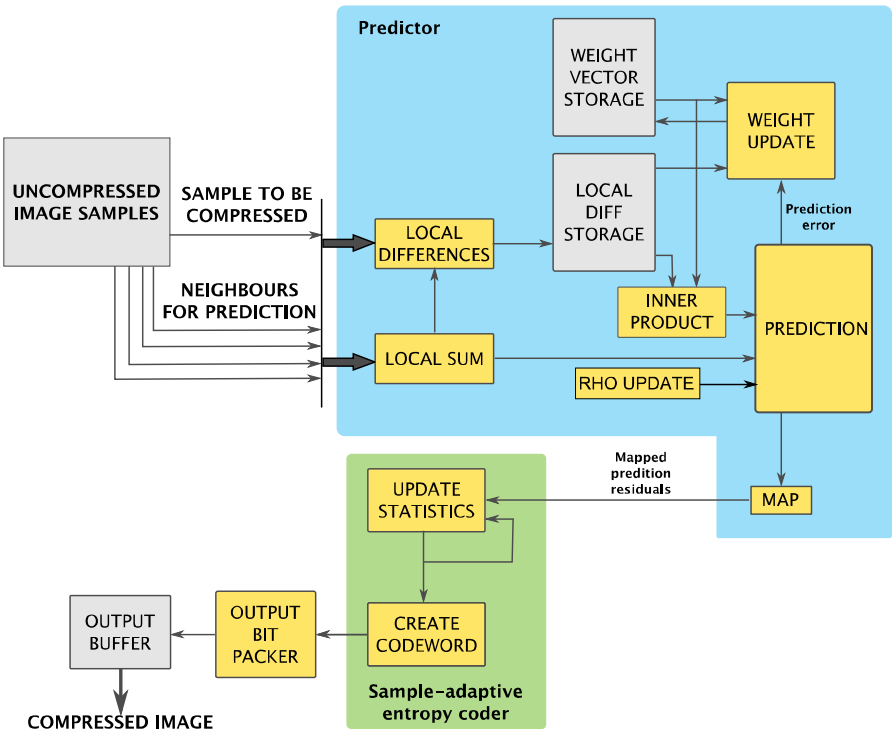


Figure 5-4: Simplified Schematic of an Implementation of the Recommended Standard with Sample-Adaptive Entropy Coding

5.4.3.3 Computation of the Predicted Central Local Difference

One of the major design efforts for a hardware implementation of the compressor is the inner product calculation needed to compute the predicted central local difference used for prediction (equation (30) of reference [1]):

$$\hat{d}_z(t) = \mathbf{W}_z^T(t)\mathbf{U}_z(t) \tag{26}$$

The main concern related to this inner product is not the number of arithmetic operations (multiplications and additions) involved, but the volume of data that has to be available when it is calculated. In particular, these data are the local difference and weight vectors. The number of elements in each vector increases with P, the user-specified parameter that determines the number of preceding spectral bands used for prediction.

In a hardware implementation, the amount of storage needed for the local difference and weight vectors, and the data dependencies that arise, are primarily affected by the order in which image samples are processed by the predictor. To illustrate this, requirements for storage and data dependencies are compared for the predictor when it processes samples in BSQ order and BIP order.

Under BSQ processing order, the following constraints are observed for a hardware implementation:

- **Storage:** In BSQ order, it is necessary to store the local difference vectors for samples in a given band in such a way that they can be used for prediction in the next band. This means storing the following set of vectors: $\boldsymbol{U}_z(0),...,U_z(t),...,U_z(N_y \times N_x - 1)$, which makes a total of $N_y \times N_x \times C_z$ local difference values.

- **Data dependencies:** The prediction and weight update operations for a sample $s_z(t)$ have to be completed before prediction can be performed for the next sample in BSQ order, $s_z(t+1)$. This is because the updated weight vector $W_z(t+1)$ has to be available to calculate $\hat{d}_z(t+1)$. As a consequence it is not possible to schedule the prediction of sample $s_z(t+1)$ in parallel with the weight update operations of sample $s_z(t)$, which may limit achievable throughput.

Under BIP order the following constraints are observed:

- **Storage:** Following prediction of a sample $s_z(t)$, only a single element of the local difference vector needs to be updated for prediction of the next sample in BIP order, $s_{z+1}(t)$. Consequently, it is only necessary to store a local difference vector, with C_z elements. However, it is necessary to ensure that, following prediction of sample $s_z(t)$, the updated weight vector $W_z(t+1)$ is available for the prediction of the next sample in the same band $s_z(t+1)$. This might imply storing the updated weight vectors in all the bands: $\boldsymbol{W}_0(t),...,W_z(t),...,W_{N_z-1}(t)$.

- **Data dependencies:** It is not necessary to complete prediction of a sample $s_z(t)$ before starting prediction of the next sample in BIP order, $s_{z+1}(t)$. This makes it possible to schedule the prediction of sample $s_{z+1}(t)$ in parallel with the weight update operation of sample $s_z(t)$.

Estimations of the amount of storage required by a specific hardware architecture is recommended to determine whether it is necessary to use additional memory external to the FPGA. Nevertheless, users can find solutions to cope with the aforementioned limitations. For instance, under BSQ order, it is possible to avoid storing the local differences if instead the compressor calculates all the elements of the local difference vector for the prediction of each sample. This approach requires the necessary neighbors to be available for the current sample to be compressed as well as those located in the P preceding bands, and it also involves computing P additional local sums and local differences. This particular solution has been adopted for software and hardware implementations described in reference [25].

5.4.3.4 Entropy Coding

The block-adaptive coding approach requires the evaluation of the different encoding options for a complete block of J samples. These evaluations require addition and comparison operations. The amount of required storage and hardware complexity does not depend on the compression order, but does depend on the number of samples in a block and the number of encoding options to be evaluated.

When encoding a mapped prediction residual using the sample-adaptive coding approach, under BSQ order it is required to have one previously processed mapped prediction residual, one accumulator, and one counter available. Under BIL or BIP orders, the same elements are needed for each band in the hyperspectral cube; i.e., N_z mapped residuals, N_z accumulator, and N_z counter values are needed.

Implementation considerations that might affect the choice between the two entropy coding approaches include:

- When sample-adaptive coding is used, compression effectiveness does not depend on sample encoding order. An implementer using the sample-adaptive coder need not be concerned with compression effectiveness considerations in selecting the sample encoding order.
- The block-adaptive coding approach leverages an existing standard, and so an implementer may be able to take advantage of an existing implementation of the block-adaptive coder.
- The sample-adaptive coding approach does not require the evaluation of multiple coding options; this may or may not provide a complexity advantage.

The operations in the sample-adaptive coding approach can be implemented with adders, shifters, and barrel shifters. The block-adaptive approach also performs a multiplication of integer variables (multiplication of two mapped prediction residuals is needed to evaluate the second-extension option). Depending on the implementation, it might be possible to avoid this multiplication, as shown by Yu *et al.* in reference [26].

5.5 DETECTOR NONUNIFORMITY CORRECTION

As discussed in 4.2.1 (see figure 4-2), nonuniformities in detector arrays can cause streaking artifacts in multispectral and hyperspectral images. The use of column-oriented local sums and reduced mode can provide improved compression effectiveness for images exhibiting such artifacts, but the elimination or reduction of such artifacts prior to compression can sometimes yield more effective compression.

Performing complete radiometric calibration onboard may be impractical, but a simpler reversible pre-processing step that reduces the severity of detector nonuniformities might be feasible. With such an approach, the pre-processing step may be applied prior to onboard compression and removed after on-the-ground decompression, obtaining exactly the same

images as if the pre-processing step was not employed. A thorough study of the possible approaches is beyond the scope of this discussion; one simple approach is illustrated as an example.

For a pushbroom imager using a separate detector element for each cross-track position (x) and spectral band (z), for each such detector element an integer offset value $b_{z,x}$ is defined. For each sample $s_{z,y,x}$ in an image, the corresponding offset value is subtracted prior to compression; i.e., the image input to the compressor is $\left\{s'_{z,y,x}\right\}_{z,y,x}$, where for each x, y, z,

$$s'_{z,y,x} = s_{z,y,x} - b_{z,x}\,. \tag{27}$$

This method is motivated by the observation that the variations in detector element characteristics are an inherent property of the detector, and there may be little change in this variation over time. It is assumed that the array of integer offset values $\{b_{z,x}\}_{z,x}$ is known to the decompressor so that this offset correction process can be reversed and thus the original image can be recovered exactly.

In practice, calibration data could be used to generate the array of offset values. For illustration purposes, a crude alternative approach is to calculate each offset value as

$$b_{z,x} = \mathrm{Round}\left[\, m_{z,x} - \mathrm{median}\left[\left\{m_{z,x}\right\}_{x=0}^{N_\mathrm{X}-1}\right]\right] \tag{28}$$

where

$$m_{z,x} = \frac{1}{N_\mathrm{Y}} \sum_{y=0}^{N_\mathrm{Y}-1} s_{z,y,x}\,. \tag{29}$$

Figures 5-5 and 5-6 show false-color images derived from raw and offset-adjusted versions of CRISM and M3 images using the method of equation (28). Figure 5-7 shows a similar result for a Hyperion image, using an offset array provided by the mission.[3] These visualizations are intended to provide some indication of the degree to which an offset adjustment can reduce the severity of streaking artifacts. Phenomena such as the artificially increased brightness on the right side of figure 5-6 (caused by the prominent dark feature in these columns of the image combined with the somewhat simplistic calculation of equation (28)) should not cause undue concern because the offset adjustment is reversible after decompression.

[3] The offset-adjusted version of the Hyperion Cuprite image is the 'Hyperion Cuprite flatfield' image included in the image corpus described in annex A.

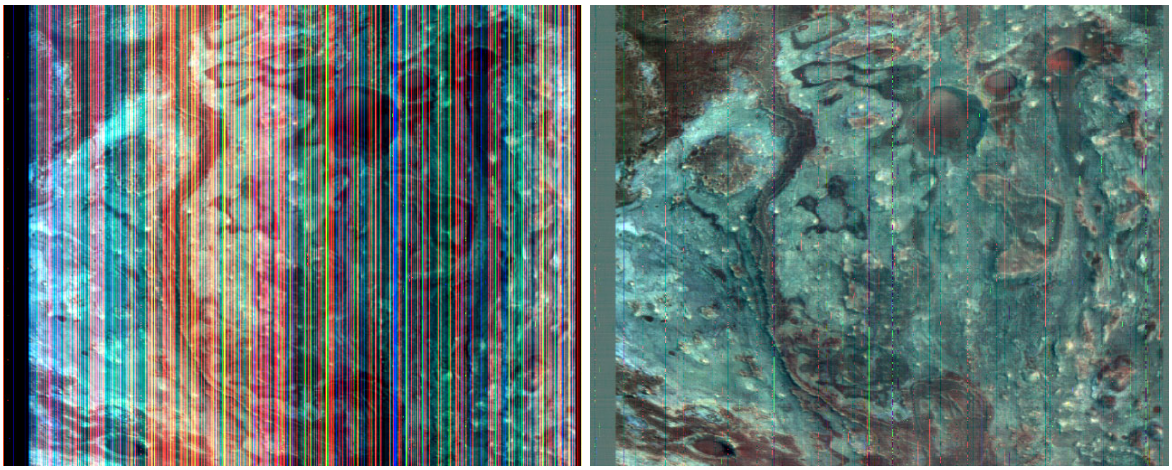


Figure 5-5: False Color Images Derived from Bands 200, 300, 400 of Raw (Left) and Offset-Adjusted (Right) Versions of CRISM FRT Image 00009326_07_sc167

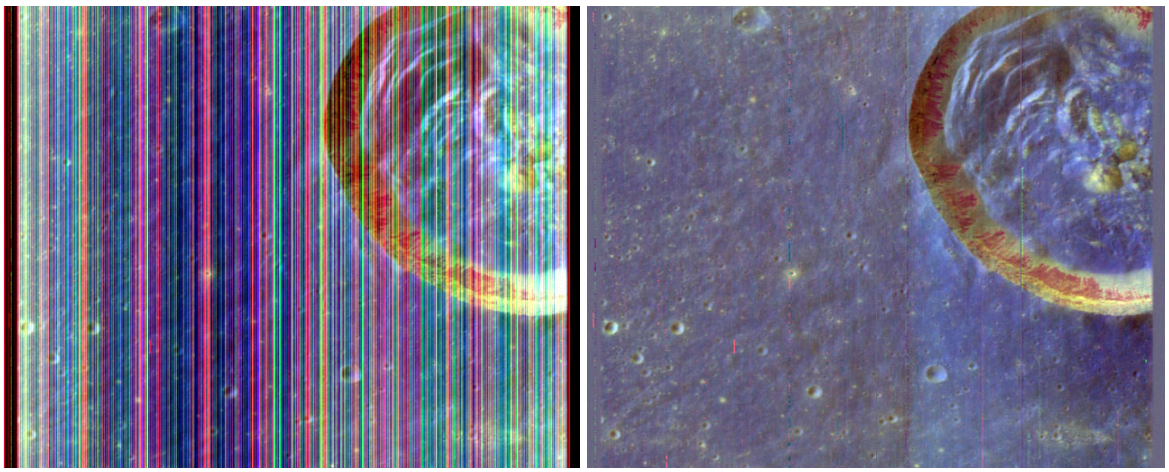


Figure 5-6: False Color Images Derived from Bands 50, 150, 200 of Raw (Left) and Offset-Adjusted (Right) Versions of M3 Target Image A

The offset adjustment dramatically decreases streaking artifacts in these three examples. But some remaining streaking artifacts are evident in the images of figures 5-5 and 5-6, including some that are clearly not well modeled as a constant additive term.

Figure 5-8 shows the compressed data rates for the raw and offset-corrected images shown in figures 5-5–5-7 for all four combinations of prediction mode and local sum type, when all other compression settings are assigned to the values indicated in annex C.

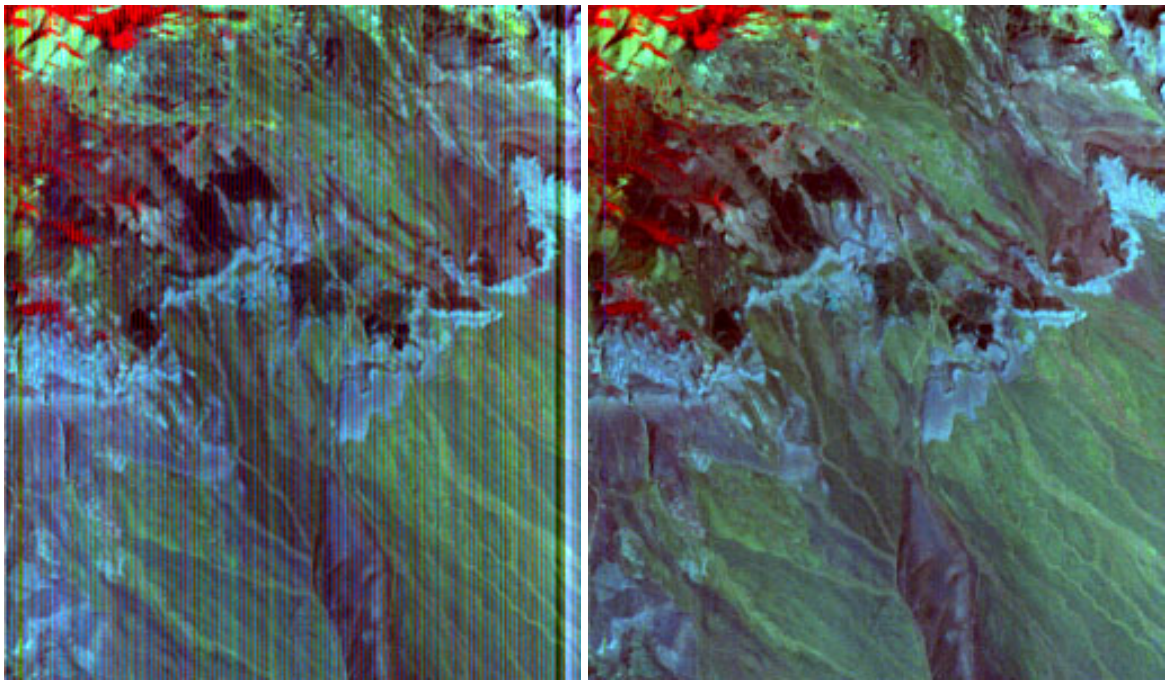


Figure 5-7: False Color Images Derived from Bands 83, 138, 200 of Raw (Left), and Offset-Adjusted (Right) Versions of Hyperion Cuprite Image

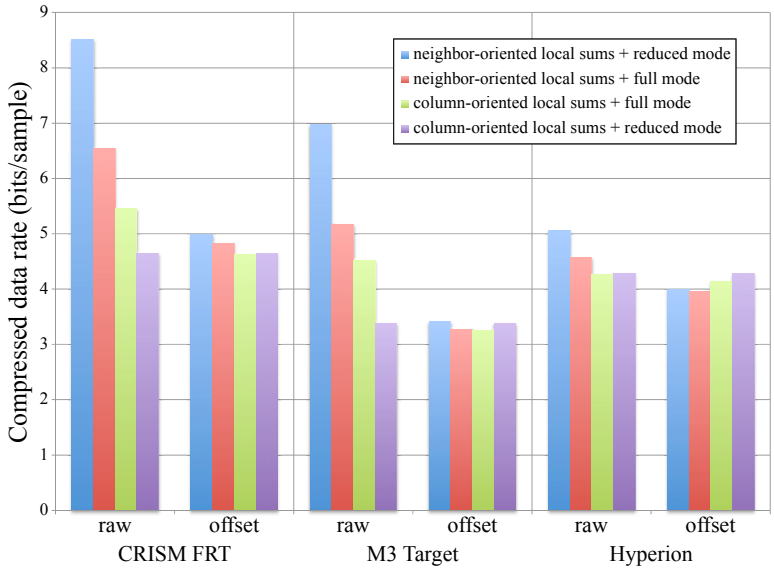


Figure 5-8: Compressed Data Rates in Bits/Sample for Raw and Offset-Corrected Images CRISM FRT 00009326_07_sc167, M3 Target A, and Hyperion Cuprite

The offset adjustment has no impact on column-oriented local sums after the first frame of the image, and consequently it has virtually no impact on compression effectiveness when column-oriented local sums are used with reduced mode. This is why, for any given image, the pair of purple bars in figure 5-8 have nearly identical height.

The compressed data rate results for the Hyperion image example demonstrate that, at least for some imagers, a noticeable improvement in compression effectiveness can be obtained by applying a pre-processing step to reduce streaking artifacts, even by a method as simple as the offset adjustment approach used here.

But pre-processing steps that reduce streaking artifacts are not guaranteed to yield an appreciable compression improvement over the simpler alternative of applying column-oriented local sums and reduced mode to the raw image, when changing values of the x index of an image correspond to changing detector elements. For example, on the CRISM image, the use of column-oriented local sums and reduced mode on the raw image yielded a compressed data rate within a half percent of the best-performing compression choice on the offset-corrected image, despite the dramatic reduction in streaking artifacts evident in figure 5-5.

Users considering image pre-processing steps to reduce detector-induced image artifacts for the sake of improving compression effectiveness should keep in mind some additional caveats:

- Some methods of correcting detector nonuniformities are not reversible. Procedures that are reversible may require the transmission of side information (such as the offset array in the method described above), and the cost to transmit such information should be considered when assessing the potential benefits.
- The offset-adjustment approach described above (as well as other approaches to nonuniformity correction) may increase the dynamic range of the image input to the compressor, and so the compressor implementation must accommodate this higher dynamic range.
- A nonuniformity correction approach that multiplies sample values by a scaling factor larger than one (as is often the case for radiometric calibration procedures) effectively increases signal energy and may actually lead to worse compression performance, even when it is effective at removing detector-induced image artifacts.

5.6 BAND RE-ORDERING

Most multi- and hyperspectral image compression algorithms follow the ‘natural’ band ordering; i.e., the different bands are compressed in order of increasing wavelength. The idea of band re-ordering (reference [27]) is that this natural ordering is not necessarily optimum in terms of compression effectiveness. That is, compression performance might be improved simply by re-arranging the order of the spectral bands in an image.

This raises the problem of finding the band ordering yielding the minimum bit-rate, which is intimately related to the compression method and can be rather complex, given the large number of possible band orderings. In most cases, the optimal adaptive on-line computation

may not be feasible onboard. Nevertheless, when the optimal ordering can be considered sensor-dependent and not image-dependent, a fixed reordering computed off-line can provide near-optimal performances (reference [27]). Depending on the compression method, the optimum ordering maximizes 'similarity' or 'complementarity' between pairs of adjacent bands according to a suitable metric. A complete overview of similarity measures and band reordering techniques is given in reference [28].

As far as this lossless compression Recommended Standard is concerned, the following remarks are in order.

- Band re-ordering is not part of the Recommended Standard, in that the Recommended Standard does not specify an algorithm to compute an improved ordering of spectral bands, nor does it provide any syntax for communicating a reordering of bands for an image. However, the standard makes no requirement that bands be arranged in order of increasing or decreasing wavelength, and so a band reordering could be applied prior to compression. From the perspective of the standard, an image with re-ordered bands is simply a different image to be compressed.

- In general, the use of band reordering has been shown to provide benefits for this lossless compression Recommended Standard (reference [28]) as well as for other lossless multispectral and hyperspectral image compressors (reference [29]). However, the performance gain is small for some imagers, and the impact of band reordering might be small when using more bands for prediction (larger values of P).

- Finding the optimum band order for a given image is a computationally intensive problem. (See reference [28] for discussions of approaches for finding good band re-orderings for different compression approaches.) However, a suboptimal ordering could still provide a worthwhile benefit over the natural order. Moreover, it is conceivable (references [27] and [28]) that one could compute a default re-ordering on the ground, and use this same default order throughout a mission phase, to provide a compression benefit with little impact on onboard computations.

6 PERFORMANCE

6.1 INTRODUCTION

This section presents compression performance results (compressed data rates in bits/sample) for the Recommended Standard as well as some other lossless compression approaches. Subsection 6.2 describes the compression methods used, and subsection 6.3 provides compression results for these compression approaches applied to the corpus of multispectral and hyperspectral images described in annex A.

For discussion purposes, images in the corpus are separated into two categories. The term 'hyperspectral' is used to refer to images in the corpus having 72 spectral bands or more, and 'multispectral' is used to refer to images in the corpus having 17 or fewer spectral bands. There are no images in the corpus having between 18 and 71 spectral bands.

Adaptive compression approaches, such as the one specified by the Recommended Standard, generally provide more effective compression on larger images. However, as described in 5.3, partitioning a larger image into independently decompressible smaller images may be desirable for the sake of limiting the impact of data loss or errors on the communications channel. Limiting input image size in this manner may also reduce implementation memory requirements for some compression approaches.

For these reasons, compression results for most compressors evaluated in this section are presented both for full images (i.e., a given image in the corpus is compressed as a single image) and segmented images (i.e., an image is first partitioned into separate smaller images that can be independently decompressed and then concatenated to form the complete larger image). For segmented images, images are partitioned along the y-axis into image 'segments' of height 32 (for hyperspectral images) or 256 (for multispectral images).

For wavelet-based image compression approaches examined, independently compressing image segments formed in this straightforward manner may cause compression efficiency to suffer noticeably. Instead, partitioning data in the transform domain may achieve a similar degree of segmentation and allow independent portions of transformed data to be efficiently compressed. However, such a segmentation defined in the transform domain does not correspond to a simple partitioning of image samples in the original image domain; whenever one segment is lost (e.g., because of data loss on a communications channel), such a loss affects adjacent segments, corrupting a limited number of samples near segment boundaries. For the wavelet-based compressors examined, 6.2.4 and 6.2.5 describe compression settings that are intended to achieve a degree of segmentation, in the transform domain, comparable to that of the other compressors.

Results included here are intended to provide some indication of the relative compression performance of different lossless compression algorithms. However, the different compressors vary significantly in complexity and it is not straightforward to provide a meaningful complexity comparison between different compressors. Some of the compression methods used here may be impractical, or at least very challenging, to implement onboard a spacecraft.

6.2 COMPRESSORS

6.2.1 OVERVIEW

This subsection lists the compressors used and describes some of the methodology used to obtain the compression results.

6.2.2 CCSDS 123.0-B-1

For the Recommended Standard, results are given for sample-adaptive coding as well as block-adaptive coding under BIL and BIP encoding orders. Results are not included for BSQ encoding order because, as discussed in 4.3.3.2, compressed data rate under BSQ encoding order is nearly identical to that obtained under BIL order. It may be recalled that under the sample-adaptive encoder, compressed bit rate does not depend on sample encoding order.

For segmented images, weight resolution $\Omega = \min\{D+4,19\}$ is used for all segments. Default weight initialization and $v_{\min} = -1$ are used for the first segment, and for all subsequent segments custom weight initialization is used with $v_{\min} = 3$, $Q = \lfloor \Omega / 2 \rfloor$, and initial weights are set equal to quantized versions of the final weights obtained from the preceding segment.

All other compression settings use the values indicated in annex C.

6.2.3 JPEG-LS

The JPEG-LS image compression standard (reference [30]) supports compression of three-dimensional ('multi-component') images such as multispectral and hyperspectral images. However, as described in 7.3.2, JPEG-LS is primarily designed to exploit two-dimensional image structure and takes almost no advantage of similarities between adjacent spectral bands. For this reason, and because of a lack of readily available JPEG-LS software implementations that support multi-component images having more than a few spectral bands, some *ad hoc* methods of applying single-component (2D) JPEG-LS compression to multispectral and hyperspectral images were considered.

The application of JPEG-LS (reference [12]) directly to the image ('direct') and also to differences between spectral bands after the first ('differential') were both evaluated. In either case, the multispectral or hyperspectral (three-dimensional) image to be compressed is converted into one or more 2D images. One could simply apply JPEG-LS independently to each spectral band (or band difference) or to each spatial-spectral slice of the 3D image. However, when each resulting 2D image is small, compression performance may be poor because small images do not allow much time for the compressor to adapt to the image, and because smaller images spend a larger fraction of compressed bits on overhead (e.g., image headers). These effects can be reduced somewhat by concatenating these 2D slices to form a single larger 2D image to be compressed. In effect, a 3D image is flattened to form a 2D image which is then compressed.

There are several different ways to flatten a 3D image into a 2D image—figure 6-1 illustrates some of them—and the method used can affect compression effectiveness, sometimes by a significant amount. Moreover, the flattening approach that yields the best performance can vary depending on the imaging instrument and can even change from image-to-image for the same imager.

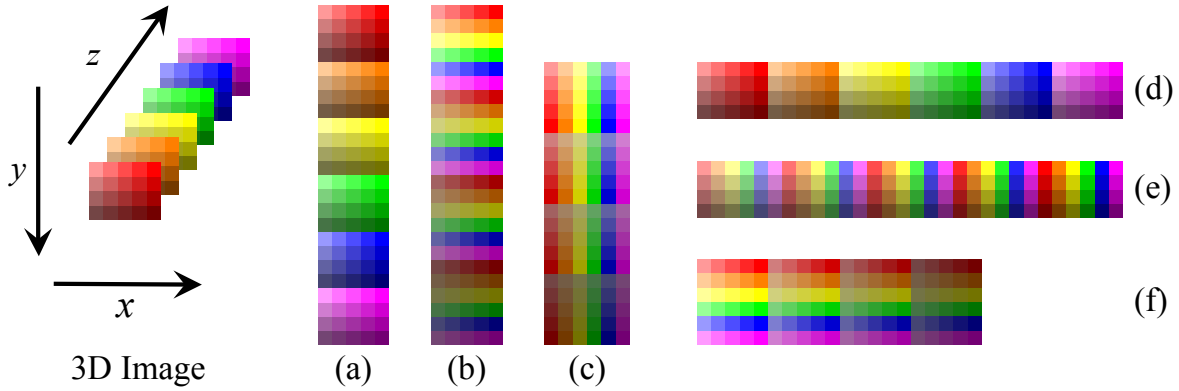


Figure 6-1: Examples of Different Methods of Forming a 2D Image by Flattening the Samples in a 3D Image

In lossless compression experiments on a subset of images from the corpus, direct compression is often, but not always, less effective than differential compression. In either case, flattening methods (a) and (f) of figure 6-1 tended to yield better performance than the others. It should be noted, however, that for an imager that produces samples in BIL or BIP order (such as a pushbroom imager), these flattening methods impose the most significant buffer constraints because JPEG-LS compresses an image in raster scan order, and these two arrangements require all samples from a band to be compressed before compressing any samples from the next spectral band. By contrast, the present Recommended Standard allows a much smaller buffer.

A thorough examination of the compression effectiveness and implementation complexity of all possible ways to using JPEG-LS to compress a 3D image is beyond the scope of this document. As an indication of compression performance that might be obtained via JPEG-LS, 6.3 includes performance results for direct and differential JPEG-LS, in both cases using flattening method (a) of figure 6-1, which corresponds to vertically concatenating spectral bands of an image.

These results were obtained using version 2.2 of the JPEG-LS implementation produced by Ismail R. Ismail and Faouzi Kossentini of the Department of Electrical and Computer Engineering, University of British Columbia (http://www.stat.columbia.edu/~jakulin/jpeg-ls/). A constant offset was added to band differences to make all of the values nonnegative prior to compression. To circumvent software limitations on the input image height supported, flattened images with height exceeding $2^{16}-1$ were split along the vertical axis as needed.

6.2.4 JPEG2000

The JPEG2000 image compression standard (reference [31]) defines a two-dimensional image coder based on wavelet transforms and bit plane encoding. The coder is designed so that progressive compression can be achieved by scanning wavelet-transformed image data multiple times, each successive scan encoding additional data that provides a more precise representation of the image, while increasing the size of the compressed image. This feature is well suited for lossy image compression, but if a sufficient number of scans are performed, it can also provide lossless compression at the expense of higher implementation complexity.

Two different wavelet transforms are available, one irreversible—meaning that the original image data cannot be exactly recovered—and one reversible, which can be exactly inverted and thus suits the purposes of this comparison. Hence, lossless compression can be achieved when a sufficiently large compressed image is allowed and the reversible wavelet transform is selected.

Three-dimensional image compression is supported in JPEG2000 by means of the so-called *Multi-Component Transform (MCT) part 2 extension*. This extension enables the coding of three-dimensional images as multiple two-dimensional images (one image for each spectral band), possibly after a one-dimensional spectral transform is employed to reduce the degree of correlation among them. Three cases of MCT operation are considered: when no spectral transform is employed ('direct'), when the reference Integer Wavelet Transform (IWT) is employed (specifically a CDF 5/3 wavelet as defined in reference [31]), and when a Pairwise Orthogonal Transform (POT) is employed (reference [32]).

To provide a relevant comparison with other compressors applied to segmented images, JPEG2000 settings are selected to provide lossless compression with a similar level of granularity when accessing data spatially. While the granularity may be obtained with the use of segments that split the image along the y-axis into multiple regions that can be decoded independently, the comparable feature in JPEG2000 is spatial scalability. To provide a comparable approach, JPEG2000 settings are adjusted so that a loss or corruption of compressed data confined to one region (segment) of the image affects only a region of similar height. The vertical distance between two regions that can be independently decoded is selected so that loss or corruption of compressed data for one region should not affect the other (assuming the bit stream can be correctly resynchronized).

Using default settings in JPEG2000 (usually 5 levels of transform, blocks of size 64×64, and appropriately defined precincts), a block in the lowest wavelet resolution defines a region of 2048 rows, which would be much larger than the segments used to produce results for other compressors. Thus appropriate values for these settings are to be determined. An additional issue is the overlap of wavelet transform information from one block to another, which will further expand this region.

In the general case, given an image of width N_X and using L levels of two-dimensional wavelet transform, the parameters that produce a comparable segment size of N are:

- Block size = $\left(\min\{N/2^L, 64\}, 4096/\min\{N/2^L, 64\}\right)$;
- Precinct size (level i=0,1...) = $\left(N \cdot 2^{-i}, N_X\right)$.

These calculations are generic regardless of the number of components transformed or whether any spectral transform is used at all. Using the calculations above, the Kakadu software implementation[4] of JPEG2000 was used to produce experimental results for equivalent segment sizes of 32 frames (for hyperspectral images) or 256 frames (for multispectral images) with 3 levels of wavelet transform. For example, for an image of 512 columns and coded with a segment size of 256 rows and 3 levels of wavelet transform, for compression using the Kakadu software, the portion of the command line input that controls these settings is:

```
Creversible=yes Cblk={32, 128} Cprecincts={256,
  512},{128,512},{64,512},{32,512}
```

Regarding the wavelet overlap, for the CDF 5/3, the overlap would be $2^L + 2^{L-1} - 1$ (in some cases a bit less), which in this example would be 11 frames.

6.2.5 CCSDS 122.0-B-1

The CCSDS 122.0-B-1 image compression standard (reference [33]) defines a two-dimensional image coder specifically designed for spaceflight implementation. It shares many features with JPEG2000, including the use of two-dimensional wavelet transforms, lossy and lossless compression capabilities, and successive quality refinements of an image during the coding procedure. However, its design was focused on meeting the constraints of spaceflight hardware. An extensive feature comparison and performance assessment is included in reference [34].

Experimental results for CCSDS 122.0-B-1 were produced using an enhanced version of the software implementation by NASA's Goddard Space Flight Center and modified by the Group on Interactive Coding of Images of the Universitat Autònoma de Barcelona that supports the use of spectral transforms. For the sake of generating comparable results, the compression performed is necessarily not strictly compliant with the CCSDS 122.0-B-1 specification in two ways. First, bit depths greater than 16 bits must be accommodated because of dynamic range expansion due to the spectral transforms. Second, segment sizes (S) less than 16 blocks are used in some cases to provide a comparable level of image segmentation. Three sets of experimental results are provided: one without any spectral transform, one with an IWT (a CDF 5/3 as defined in reference [31]), and one with a POT (reference [32]).

[4] Kakadu Software was developed by D. Taubman, who provided an online demonstration version.

The coder was configured to produce lossless bit streams by setting compression parameters as follows: SegByteLimit=2^{27}, DCStop=0, BitPlaneStop=0, StageStop=Stage 4, and UseFill=0. Optimal variable-length code selections were employed (OptDCSelect=1, OptACSelect=1), and codeword length was set to 8 bits. No custom wavelet weights were employed.

To provide results using the previously stated segment sizes, the following settings were employed. For segmented hyperspectral images, the number of blocks per segment, *S*, was assigned so that segments would cover a region of 8 frames of coefficients after a wavelet transform (i.e., $S = \lceil N_x / 8 \rceil$). The Integer Wavelet employed by CCSDS 122.0-B-1 for two-dimensional image coding is an Integer 9/7M wavelet, which has 21 frames of worst-case error propagation in both directions. The described approach has an equivalent segment size of 51 frames (see subsection 2.4.3.3 of reference [34]), which exceeds the 32-frame segment initially required.

Similarly, to obtain a comparable configuration for segmented multispectral images, the number of blocks per segment was set to cover a region of 256 frames of coefficients after the wavelet transform (i.e., $S = 32\lceil N_X / 8 \rceil$). However, this second configuration was not expected to deviate substantially from the former one, as the bit-rate increase produced by increasing the value of *S* is minimal at high rate (see subsection 3.5 of [34]).

6.2.6 LUT PREDICTOR

The LookUp-Table (LUT) compressor (reference [35]) is a single-pass sequential predictive lossless compression algorithm designed for hyperspectral images. The prediction module of the LUT compressor has very low complexity, but prediction residuals are encoded using an adaptive range coding approach that may have relatively high implementation complexity compared to the entropy coding methods included in the present Recommended Standard.

For this reason, 6.3 includes compression results for the LUT predictor combined with mapping of prediction residuals to nonnegative integers (in a manner similar to that described in 3.2.4) followed by lossless encoding using the sample-adaptive entropy coder. Reference [35] does not describe how the LUT compressor performs prediction in the first band of an image. For the results included here, the predicted value of a sample in the first band is equal to the neighboring sample above, except in the first row, where it is equal to the left neighbor.

6.2.7 ESA COMPRESSOR

The compressor proposed by ESA (reference [29]) targets very low complexity. The algorithm design is based on the block-based predictor first proposed in reference [36]. This predictor is applied to nonoverlapping spatial blocks of 16×16 samples. Each block is predicted from the co-located block in the previous band using a least mean squares technique. An individual linear predictor is calculated for all samples of each block, and its

parameters (offset and gain) are encoded in the compressed image. The mapped prediction residuals are encoded by employing a sample-adaptive Golomb coder, where the code parameter may be restricted to be a power of two. Except for minor differences in the mapping and Golomb parameter adaptation procedure, the entropy coding stage is essentially equivalent to the sample-adaptive coding option included in the present Recommended Standard. The complete algorithm, as well as some extensions including near-lossless compression and band ordering, are described in reference [29].

Compression results here make use of the ESA predictor along with the sample-adaptive entropy coding method formalized in the standard.

6.3 RESULTS

6.3.1 FULL IMAGE VS. SEGMENTED IMAGE COMPRESSION

As described previously in this section and also in 5.3, partitioning a larger image into independently decompressible smaller images may be desirable to limit the impact of data loss and to reduce implementation memory requirements for some compression approaches. However, this partitioning also reduces compression effectiveness in most cases (see 5.3.2) because of the additional overhead cost of image headers, as well as other considerations like boundary handling or poorer adaptive prediction for predictive methods.

Figure 5-4 illustrates this reduction in compression effectiveness; as expected, higher efficiency is obtained for larger image segment sizes. The same effect can be observed in table 6-1. For hyperspectral images, full image compression always performs as well as or better than segmented compression. For multispectral images, where segment size is selected to be larger, this performance penalty is less apparent, and in some cases segmentation even provides a small benefit.

For clarity, only one configuration is presented in table 6-1 for each algorithm; results for other configurations are consistent with those presented below. Tables with complete results can be found in annex E.

Table 6-1: Average Compressed Data Rates (in Bits/Sample) for Full Image and Segmented Compression

		CCSDS-123.0-B		JPEG-LS direct		ESA + sample-adaptive coding		LUT + sample-adaptive coding		CCSDS-122.0-B		JPEG2000	
	Instrument	Segmented	Full Image	Segmented	Full Image	Segmented	Full Image	Segmented	Full Image	Segmented	Full Image	Segmented	Full Image
Hyperspectral Images	IASI	4.77	4.75	6.61	6.60	4.90	4.91	5.72	5.57	7.62	7.55	7.42	7.02
	AIRS	4.32	4.30	6.35	6.35	4.66	4.68	6.13	5.66	6.91	6.86	7.01	6.62
	CRISM-FRT	5.21	5.06	5.74	5.53	8.92	8.92	9.60	9.54	6.93	6.92	6.30	5.97
	CRISM-HRL	4.73	4.56	5.75	5.55	8.38	8.38	9.15	9.06	6.82	6.80	6.32	5.95
	CRISM-MSP	2.90	2.55	3.75	3.42	6.80	6.80	7.32	7.02	5.01	4.91	4.70	3.85
	M3-Global	2.37	2.14	4.93	4.83	6.45	6.45	7.30	7.20	5.74	5.72	5.40	5.10
	M3-Target	3.25	3.09	3.82	3.66	6.60	6.60	7.51	7.44	5.03	5.02	4.29	4.00
	Hyperion	4.37	4.30	5.09	5.02	5.52	5.52	6.16	6.08	5.60	5.58	5.39	5.14
	Hyperion flatfield	3.98	3.97	4.81	4.80	4.11	4.11	4.55	4.54	5.00	4.99	5.08	4.89
	SFSI	4.69	4.67	4.77	4.75	4.91	4.91	5.43	5.43	5.30	5.30	4.79	4.65
	SFSI_rmnoise	3.01	2.96	4.40	4.35	4.07	4.07	4.80	4.70	4.89	4.88	4.56	4.42
	AVIRIS(16-bit raw)	5.99	5.98	8.64	8.61	6.16	6.16	7.42	6.87	8.79	8.78	8.98	8.88
	AVIRIS(12-bit raw)	2.69	2.68	4.56	4.54	3.01	3.01	3.53	3.44	4.73	4.72	4.67	4.59
	AVIRIS(16-bit cal)	3.76	3.74	6.41	6.39	4.32	4.32	4.75	4.54	6.58	6.57	6.65	6.56
	CASI	5.03	5.02	6.79	6.77	5.10	5.10	5.96	5.65	7.04	7.02	7.09	6.97
Multispectral Images	MODIS-night	4.64	4.70	5.39	5.38	7.76	7.76	7.24	7.22	6.20	6.20	5.54	5.51
	MODIS-day	5.63	5.72	4.69	4.69	5.75	5.75	6.54	6.44	5.59	5.59	5.43	5.37
	MODIS-500m	7.19	7.20	7.63	7.62	8.36	8.36	8.88	8.88	8.21	8.22	7.80	7.77
	MODIS-250m	6.43	6.48	6.96	6.96	7.15	7.15	7.73	7.73	7.14	7.14	7.11	7.08
	MSG	3.40	3.39	3.50	3.50	4.11	4.11	5.07	5.06	3.70	3.70	3.53	3.50
	LANDSAT	3.37	3.37	3.70	3.70	3.91	3.91	4.32	4.32	4.03	4.03	3.90	3.87
	PLEIADES	7.32	7.32	7.85	7.84	7.95	7.95	8.74	8.65	8.05	8.05	8.25	8.15
	VEGETATION_level1b	5.26	5.26	5.31	5.30	5.86	5.86	6.97	6.96	5.59	5.59	5.53	5.48
	VEGETATION_level1c	5.04	5.04	5.27	5.26	5.55	5.55	6.77	6.77	5.46	5.46	5.46	5.42
	SPOT5	4.53	4.53	4.71	4.71	5.17	5.17	5.66	5.66	4.96	4.96	4.94	4.92

NOTE – Better performance is indicated in green.

Figure 6-2 focuses on the comparison of both modes (full image vs. segmented) for the Recommended Standard with the default compression settings. A significant penalty can be observed for some hyperspectral images because the selected segment size is small. In contrast, when larger segments are used, as in the multispectral images, both compression modes perform similarly and even a slight improvement can occur in some cases for segmented compression.

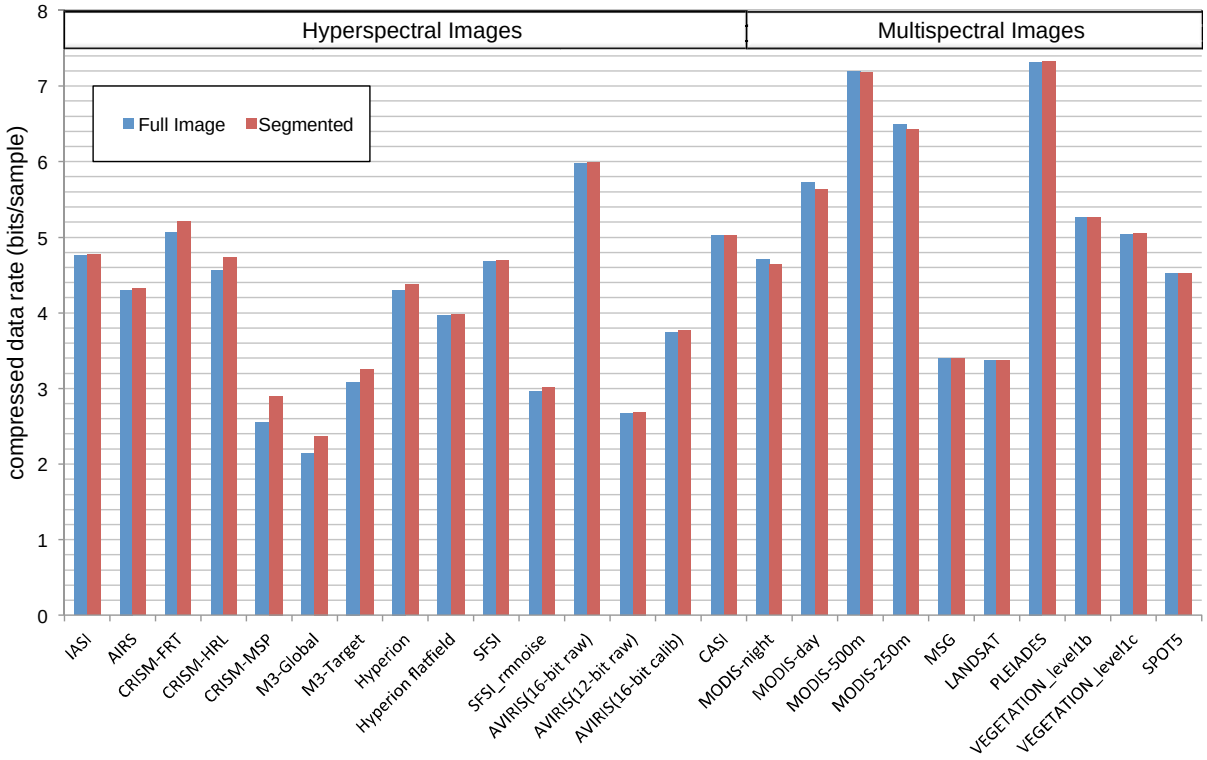


Figure 6-2: Comparison of Average Compressed Data Rates (in Bits/Sample) for the Recommended Standard for Full Image And Segmented Cases

6.3.2 FULL IMAGE COMPRESSION

Average compressed data rates for full image compression are given in table 6-2 for a larger set of compression approaches than presented in table 6-1 of 6.3.1.

Table 6-2: Average Compressed Data Rates (in Bits/Sample) for Full Image Compression

	Instrument	CCSDS-123.0-B	JPEG-LS		ESA + sample-adaptive coding	LUT + sample-adaptive coding	CCSDS-122.0-B			JPEG2000		
			direct	differential			direct	+ IWT	+ POT	direct	+ IWT	+ POT
Hyperspectral Images	IASI	4.75	6.60	5.02	4.91	5.57	7.55	5.63	6.11	7.02	5.21	5.55
	AIRS	4.30	6.35	4.82	4.68	5.66	6.86	5.04	5.03	6.62	4.78	4.67
	CRISM-FRT	5.06	5.53	5.48	8.92	9.54	6.92	6.95	7.00	5.97	5.86	6.08
	CRISM-HRL	4.56	5.55	4.94	8.38	9.06	6.80	6.41	6.52	5.95	5.32	5.58
	CRISM-MSP	2.55	3.42	2.99	6.80	7.02	4.91	4.72	5.25	3.85	3.54	4.02
	M3-Global	2.14	4.83	2.56	6.45	7.20	5.72	4.29	4.50	5.10	3.13	3.56
	M3-Target	3.09	3.66	3.33	6.60	7.44	5.02	4.60	4.70	4.00	3.50	3.64
	Hyperion	4.30	5.02	4.59	5.52	6.08	5.58	5.05	5.07	5.14	4.52	4.58
	Hyperion flatfield	3.97	4.80	4.33	4.11	4.54	4.99	4.39	4.32	4.89	4.19	4.05
	SFSI	4.67	4.75	5.02	4.91	5.43	5.30	4.97	4.90	4.65	4.61	4.56
	SFSI_rmnoise	2.96	4.35	3.08	4.07	4.70	4.88	3.92	3.75	4.42	3.34	3.23
	AVIRIS(16-bit raw)	5.98	8.61	6.67	6.16	6.87	8.78	6.82	6.46	8.88	6.79	6.38
	AVIRIS(12-bit raw)	2.68	4.54	3.32	3.01	3.44	4.72	3.57	3.33	4.59	3.34	3.04
	AVIRIS(16-bit cal)	3.74	6.39	4.46	4.32	4.54	6.57	4.58	4.41	6.56	4.42	4.21
	CASI	5.02	6.77	5.33	5.10	5.65	7.02	5.46	5.56	6.97	5.28	5.37
Multispectral Images	MODIS-night	4.70	5.38	5.84	7.76	7.22	6.20	6.44	5.15	5.51	5.78	4.98
	MODIS-day	5.72	4.69	5.03	5.75	6.44	5.59	6.65	6.52	5.37	6.50	6.28
	MODIS-500m	7.20	7.62	6.64	8.36	8.88	8.22	8.16	7.56	7.77	7.18	6.95
	MODIS-250m	6.48	6.96	6.62	7.15	7.73	7.14	7.07	6.50	7.08	6.99	6.41
	MSG	3.39	3.50	3.83	4.11	5.06	3.70	4.06	3.90	3.50	3.87	3.70
	LANDSAT	3.37	3.70	3.62	3.91	4.32	4.03	3.87	3.81	3.87	3.67	3.60
	PLEIADES	7.32	7.84	7.31	7.95	8.65	8.05	7.58	7.83	8.15	7.66	7.92
	VEGETATION_level1b	5.26	5.30	5.05	5.86	6.96	5.59	5.54	5.53	5.48	5.39	5.36
	VEGETATION_level1c	5.04	5.26	5.02	5.55	6.77	5.46	5.35	5.30	5.42	5.31	5.25
	SPOT5	4.53	4.71	4.71	5.17	5.66	4.96	5.07	4.70	4.92	5.03	4.61

NOTE – Better performance is indicated in green.

Of the algorithms considered, the Recommended Standard typically delivers the best average lossless compression performance for full image compression using the test images. In particular, it provides the best results for all of the hyperspectral image data except SFSI, where the combination of JPEG2000 and POT performs slightly better, but at the cost of significantly higher complexity and memory requirements. For the multispectral images, the Recommended Standard often provides the best average lossless results, but in some cases is outperformed by JPEG-LS (alone or with a pre-processor) or the combination of JPEG2000 and POT, particularly for MODIS images.

Complete results for other compression settings can be found in annex E.

6.3.3 SEGMENTED IMAGE COMPRESSION

Average compressed data rates for compression of segmented images are given in table 6-3 for a larger set of compression approaches than presented in table 6-1 of 6.3.1.

Table 6-3: Average Compressed Data Rates (in Bits/Sample) for Segmented Image Compression

	Instrument	CCSDS-123.0-B	JPEG-LS		ESA + sample-adaptive coding	LUT + sample-adaptive coding	CCSDS-122.0-B			JPEG2000		
			direct	differential			direct	+ IWT	+ POT	direct	+ IWT	+ POT
Hyperspectral Images Segment height = 32	IASI	4.77	6.61	5.03	4.91	5.72	7.62	5.70	6.18	7.42	5.64	6.02
	AIRS	4.32	6.35	4.85	4.68	6.13	6.91	5.09	5.07	7.01	5.19	5.14
	CRISM-FRT	5.21	5.74	5.73	8.92	9.60	6.93	6.95	7.01	6.30	6.22	6.40
	CRISM-HRL	4.73	5.75	5.21	8.38	9.15	6.82	6.43	6.54	6.32	5.75	5.98
	CRISM-MSP	2.90	3.75	3.47	6.80	7.32	5.01	4.82	5.35	4.70	4.45	4.95
	M3-Global	2.37	4.93	2.91	6.45	7.30	5.74	4.30	4.52	5.40	3.57	3.94
	M3-Target	3.25	3.82	3.56	6.60	7.51	5.03	4.61	4.70	4.29	3.85	3.98
	Hyperion	4.37	5.09	4.69	5.52	6.16	5.60	5.07	5.08	5.39	4.80	4.86
	Hyperion flatfield	3.98	4.81	4.34	4.11	4.55	5.00	4.40	4.33	5.08	4.40	4.27
	SFSI	4.69	4.77	5.03	4.91	5.43	5.30	4.98	4.91	4.79	4.72	4.67
	SFSI_rmnoise	3.01	4.40	3.16	4.07	4.80	4.89	3.93	3.76	4.56	3.51	3.38
	AVIRIS(16-bit raw)	5.99	8.64	6.71	6.16	7.42	8.79	6.82	6.47	8.98	6.87	6.46
	AVIRIS(12-bit raw)	2.69	4.56	3.34	3.01	3.53	4.73	3.57	3.33	4.67	3.41	3.13
	AVIRIS(16-bit cal)	3.76	6.41	4.49	4.32	4.75	6.58	4.58	4.42	6.65	4.50	4.30
	CASI	5.03	6.79	5.36	5.10	5.96	7.04	5.47	5.56	7.09	5.40	5.49
Multispectral Images Segment height = 256	MODIS-night	4.64	5.39	5.85	7.76	7.24	6.20	6.44	5.15	5.54	5.80	5.01
	MODIS-day	5.63	4.69	5.04	5.75	6.54	5.59	6.65	6.52	5.43	6.57	6.32
	MODIS-500m	7.19	7.63	6.66	8.36	8.88	8.21	8.16	7.56	7.80	7.16	6.95
	MODIS-250m	6.43	6.96	6.63	7.15	7.73	7.14	7.07	6.50	7.11	7.02	6.43
	MSG	3.40	3.50	3.83	4.11	5.07	3.70	4.06	3.90	3.53	3.90	3.72
	LANDSAT	3.37	3.70	3.62	3.91	4.32	4.03	3.87	3.80	3.90	3.69	3.62
	PLEIADES	7.32	7.85	7.33	7.95	8.74	8.05	7.58	7.83	8.25	7.75	8.01
	VEGETATION_level1b	5.26	5.31	5.06	5.86	6.97	5.59	5.54	5.53	5.53	5.45	5.41
	VEGETATION_level1c	5.04	5.27	5.03	5.55	6.77	5.46	5.35	5.29	5.46	5.34	5.28
	SPOT5	4.53	4.71	4.72	5.17	5.66	4.96	5.07	4.70	4.94	5.06	4.64

NOTE – Better performance is indicated in green.

Performance comparisons between different compression approaches for segmented images are similar to the full image case. Among the compression approaches evaluated, the Recommended Standard provides the lowest compressed bit rate for all hyperspectral data in the corpus except SFSI, for which the combination of JPEG2000 and POT provides more effective compression but at a cost of substantially higher complexity. For multispectral images, the Recommended Standard performs well but is sometimes outperformed by one of the JPEG-LS approaches evaluated.

7 STANDARD DEVELOPMENT CONSIDERATIONS

7.1 OVERVIEW

This section describes some of the decisions made in defining the Recommended Standard along with some of the motivation behind those decisions. Subsection 7.2 provides an overview of the algorithm definition process and documents the motivation and analysis that led to the selection of this algorithm. Subsection 7.3 describes differences between the Recommended Standard and two well-known image compression standards.

7.2 SELECTION PROCESS

In 2007, the CCSDS Multispectral & Hyperspectral Data Compression Working Group was established to develop data compression recommendation(s) suitable for space-borne compression of multispectral and hyperspectral imagery. It was the assumption of the Working Group that proposed algorithms would fall into one or more of the following categories:

a) lossless compressors;

b) lossy compressors providing adjustable coded data rate or reconstructed data quality;

c) near-lossless compressors.

The Working Group agreed that its first priority for standardization would be lossless compression, which is the subject of the present Recommended Standard.

The Working Group established requirements, listed in table 7-1, that a compressor (whether lossy or lossless) must satisfy to be considered suitable for standardization; these requirements reflect the envisioned application of real-time hardware compression onboard a spacecraft.

Table 7-1: Image Compression Requirements

1	A free license must be available for use of the compressor by CCSDS member agencies.
2	The compressor must accommodate instrument bit depths from 8 to 16 bits per sample.
3	The compressor must be capable of providing real-time hardware compression (20 Msamples/sec throughput @ ≤ 0.5 W/Msamples/sec using 2008 space electronics technology).
4	The compressor must be capable of scan-based and frame-based operation.
5	The compressor must limit the effects of a packet loss due to bit errors in transmission to a small region of the data.
6	The compressor must achieve effective compression performance without post-launch adjustment or updates of compression parameters.

In addition to these mandatory requirements, the Working Group also established selection criteria for the evaluation of proposed compressors, given in annex D. In summary, the selection criteria, as applied to candidate lossless compressors, are:

a) *implementation complexity*, or the ease with which a compressor can be made to provide high speed compression in a hardware implementation;

b) *compression effectiveness* measured by the compressed data rate required to losslessly compress a given multispectral or hyperspectral image;

c) *flexibility*, which could include features such as the ability to provide both lossless and lossy compression from the same algorithm, or the ability to improve compression effectiveness by adjusting compression parameters;

d) *error containment*, or the ability to effectively limit the impact of a communications packet loss on the reconstructed image;

e) *robustness*, or the ability of a compressor to provide effective compression even when the input image suffers from detector defects (e.g., dead pixels) or instrument defects (e.g., spatial and spectral misregistration, keystoning);

f) *user-friendliness*, or a compressor's ability to provide effective compression without requiring significant expertise on the part of the user;

g) *license considerations*, insofar as a third party will have more incentive to develop an implementation of the compressor if it can be used by both CCSDS member agencies as well as commercial customers without requiring a license fee.

The working group also assembled a diverse set of multispectral and hyperspectral test images covering a variety of space-borne imaging sensors, including sounders, imaging spectrometers, optical mid- and high-resolution sensors. The image test set is described in annex A.

Candidate algorithms were proposed and evaluations were conducted based on the selection criteria described above. In addition, implementation architecture studies were performed to assess the real-time processing potential of proposed algorithms, which is a key consideration for spacecraft applications. Lossless compression algorithms evaluated as part of the process of developing the Recommended Standard included the JPEG2000, JPEG-LS, and CCSDS 122.0-B-1 image compression standards, compression algorithms defined in references [37] and [38], as well as algorithms proposed by NASA (reference [18]) and ESA (reference [29]). Section 6 includes results illustrating compression effectiveness of several of these compressors, and 7.3 below discusses the JPEG2000 and JPEG-LS standards.

The working group also evaluated the LUT compression approach of reference [35] as part of its assessment (see 6.2.6). The LUT approach was appealing because of its low complexity predictor and outstanding compression performance on the 1997 AVIRIS sample images which have been widely used in the literature for assessing the effectiveness of hyperspectral image compressors. However, in reference [39], it was demonstrated that the 1997 AVIRIS sample images include calibration-induced data artifacts that were being effectively exploited

by the adaptive range encoder being used by the LUT compressor. The LUT compressor provided noticeably less effective compression than the FL compressor on images that did not exhibit such regularities (see reference [39]) and also when the LUT range coder is replaced by a lower complexity approach. Consequently, the working group did not further pursue standardization of a LUT-based compressor.

Implementation complexity played a significant role in the final algorithm selection. After some initial assessments of the various compressors, the working group narrowed the selection to the compressors proposed by ESA and NASA, motivated in large part by the fact that both approaches are amenable to a high-speed hardware implementation onboard a spacecraft.

Both the NASA and ESA proposed compressors consist of a predictor followed by entropy coding of prediction residuals. To first order, both predictors have similar computational complexity, and both entropy coders have similar complexity and compression effectiveness. Since either predictor could be used with either entropy coder, the Working Group ultimately focused its attention on a comparison of compression effectiveness results for the two proposed predictors when used with the same sample-adaptive encoder described in 3.3.4. In this comparison, the FL predictor proposed by NASA demonstrated compression effectiveness advantages on hyperspectral images containing streaking artifacts and on multispectral images.

The working group considered the possibility of augmenting the sample-adaptive encoder to incorporate run length (or similar) coding to provide more effective compression for long sequences of perfectly predicted samples. Ultimately it was the consensus of the working group that large regions of saturated or all-zero samples were not sufficiently common in practice to warrant this added complication.

A consensus was reached in 2010 to standardize the FL predictor along with the block-adaptive and sample-adaptive coding options. The resulting image compression Recommended Standard is the algorithm initially proposed by NASA, with the additional possibility of using the previously standardized CCSDS 121.0-B-2 block-adaptive encoding method (reference [13]) as an alternative to the new sample-adaptive encoder. Subsection 4.3.4 describes some of the differences between the two encoding methods which motivated the working group to retain both options as part of the standard. The chosen algorithm provides good compression performance for both hyperspectral and multispectral imagers and low complexity.

7.3 COMPARISON WITH OTHER IMAGE COMPRESSION STANDARDS

7.3.1 INTRODUCTION

A standard for multispectral and hyperspectral (three-dimensional) lossless image compression could have been developed by building on an existing two-dimensional image compression standard, rather than developing a new standard. Candidates for such an approach include the wavelet-based image compression standards JPEG2000 and

CCSDS 122.0-B-1, and the predictive-based JPEG-LS standard, all of which are capable of providing lossless image compression.

However, wavelet-based compression approaches require significantly higher computational complexity than the sequential predictive approach ultimately standardized, while providing little or no improvement in compression effectiveness in return.

The JPEG-LS compression standard has low complexity, comparable to that of the present Recommended Standard. However, JPEG-LS was not designed to exploit the three-dimensional structure present in multispectral and hyperspectral images, and simple modifications to exploit some of this structure via a pre-processing step prior to JPEG-LS compression failed to yield compression effectiveness matching that of the Recommended Standard.

Comparisons between the Recommended Standard and the JPEG-LS and JPEG2000 image compression standards are further detailed in the following subsections.

7.3.2 JPEG-LS

JPEG-LS (reference [30]) is an International Organization for Standardization (ISO) and International Telecommunication Union (ITU) standard for lossless and near-lossless compression of continuous-tone still images. It provides more effective lossless compression than the lossless JPEG standard (reference [12]), while maintaining low complexity. JPEG-LS is based on the LOCO-I (Low Complexity Lossless Compression for Images) algorithm (reference [12]).

The near-lossless option included in JPEG-LS allows lossy compression where the maximum reconstruction error is bounded by a user-specified parameter, trading reconstructed image fidelity for compression ratio. This capability is not included as part of the current Recommended Standard, though it would be feasible to add it as a future extension. Hereinafter, JPEG-LS is discussed in the context of lossless compression.

FL and LOCO-I are both low-complexity one-pass sequential predictive image compressors that adaptively predict each sample value based on preceding samples in the image and then losslessly encode the differences between predicted and actual sample values. Both adaptively update statistics used in prediction and entropy coding during the course of compression.

The LOCO-I predictor is nonlinear, incorporates a simple edge-detection approach to vary predictions when horizontal or vertical edges are likely, and alters the prediction depending on contexts defined by causal neighbors; statistics for each context are adaptively updated during compression. The FL predictor uses an adaptive linear filtering approach to perform prediction, and does not incorporate context modeling or any direct edge detection method as part of its prediction.

FL incorporates options (column-oriented local sums and reduced mode) to provide better compression performance on images containing pushbroom streaking artifacts (see 4.2.1). JPEG-LS was not specifically designed to handle such artifacts, although the simple edge-detection incorporated as part of its prediction may provide some robustness to them.

The JPEG-LS entropy coder incorporates a 'run mode' that allows efficient coding of long sequences of identically valued samples. The block-adaptive entropy coding option incorporates a similar feature (via the 'zero-block' coding described in reference [13]), but the sample-adaptive coding option does not. Apart from this difference, the sample-adaptive coder is very similar to the JPEG-LS entropy coder.

The JPEG-LS standard supports 'multi-component' (3D) images, and the image components (i.e., spectral bands) need not all have the same dimensions. JPEG-LS multi-component compression allows encoding in 'non-interleaved', 'line-interleaved', or 'sample-interleaved' orders, which correspond to BSQ, BIL, and BIP orders, respectively, when all components have the same dimensions.

In multi-component JPEG-LS compression, counters used to track context statistics may be shared between bands (depending on the encoding order), but otherwise JPEG-LS takes no advantage of similarities between spectral bands. This is the most significant difference between JPEG-LS and the present Recommended Standard: the algorithm underlying JPEG-LS is essentially a 2D compressor, while the FL compressor significantly exploits 3D structure by using a small three-dimensional neighborhood for prediction.

One can often improve the compression effectiveness of JPEG-LS on multispectral and hyperspectral images by first applying an invertible transformation (such as taking differences between consecutive spectral bands) and then flattening the 3D image to form a 2D image to be compressed by JPEG-LS, as discussed in 6.2.3. Approaches along these lines were evaluated by the Working Group but they provided less effective compression than the present Recommended Standard on the test images in annex A. Moreover, as discussed in 6.2.3, the best performing methods of flattening the image would also impose the most significant buffer constraints for pushbroom imagers which naturally produce samples in BIP or BIL order.

7.3.3 JPEG2000

The JPEG2000 image compression standard relies on progressive bit-plane encoding of wavelet-transformed image data and provides a choice of integer and floating-point Discrete Wavelet Transforms (DWTs) so that effective lossy and lossless compression can be achieved. Moreover, for multicomponent images, the MCT extension defined in Part 2 of JPEG2000 offers the possibility of using a spectral transform prior to two-dimensional JPEG2000 compression of the decorrelated bands. For lossy coding, Lagrangian rate-control can be performed, but this is not necessary for lossless coding.

The use of context modeling combined with arithmetic coding and Lagrangian rate-control leads to high compression effectiveness. Some of the components of JPEG2000 that help to

provide high compression performance (context modeling, arithmetic coding, Lagrangian rate-distortion optimization) have high implementation complexity, and this limits the suitability of JPEG2000 for space-borne missions, which require high data-throughput rates and have limited capacity of acquisition, storage, and transmission. Using JPEG2000 with the MCT option results in an even higher complexity.

For the performance assessment, the Working Group computed lossless data rates for the test data set obtained using JPEG2000 with and without spectral transforms. To make a fair comparison between proposed algorithm and JPEG2000, under similar resiliency conditions, the settings of the JPEG2000 implementation were adjusted as described in 6.2.4. Such comparisons are detailed in 6.3. The best performance was obtained by employing the POT for spectral decorrelation before applying JPEG2000 on each decorrelated band. Even so, there are only a few images in the corpus for which this approach slightly outperformed the Recommended Standard.

JPEG2000 includes capabilities that the Recommended Standard lacks. Not only can it provide lossy compression, it can also be used to provide Region-Of-Interest (ROI) coding; i.e., certain regions of an image can be encoded with higher fidelity than remaining portions of the image. JPEG2000 allows compressed image data to be arranged so that image resolution progressively improves as more compressed data are received.

JPEG2000 code-blocks are independently encoded, and thus compression is amenable to parallelization at the code-block level. Independent coding of JPEG2000 code-blocks and resynchronization markers also offer error-containment features in the event of data loss or bit errors.

As a conclusion, JPEG2000-MCT offers a lot of benefits, including random image data access, and flexibility in arranging progression order, but with a high implementation cost. The proposed Recommended Standard targets lossless compression of multispectral and hyperspectral images with low complexity. It offers fewer features but the coding performance is usually better than that of the best configuration of JPEG2000.

ANNEX A

TEST IMAGES

Table A-1 summarizes the corpus of hyperspectral and multispectral images used for compression testing and evaluation in the course of developing the Recommended Standard.

Table A-1: Summary of the Corpus of Hyperspectral and Multispectral Test Images

Instrument	**Image Type**	**Bit Depth, *D***	N_Z	N_X	N_Y
IASI	calibrated	12	8461	66	60×4
AIRS	raw	12–14	1501	90	135×10
CRISM	FRT, raw	12	545	640	{420×4, 450, 480×2, 510×2}
CRISM	HRL, raw	12	545	320	{420, 450×2, 480}
CRISM	MSP, raw	12	74	64	2700×7
M3	target, raw	12	260	640	{1774, 2386, 2843}
M3	global, raw	12	86	320	{11935, 28283}
Hyperion	raw	12	242	256	{1024, 3187, 3176, 3242}
SFSI	raw	12	240	496	140
AVIRIS	16-bit, raw	16	224	680	512×5
AVIRIS	12-bit, raw	12	224	{614, 680}	512
AVIRIS	16-bit, calibrated	16	224	677	512×5
CASI	raw	12	72	405	{2852, 1225}
MODIS	night, raw	12	17	1354	2030×5
MODIS	day, raw	12	14	1354	2030×5
MODIS	500m, raw	12	5	2708	4060×5
MODIS	250m, raw	12	2	5416	8120×5
MSG	calibrated	10	11	3712	3712×3
Landsat	raw	8	6	1024	1024×3
PLEIADES	HR, simulated	12	4	224	{2456, 3928, 2448×4}
Vegetation	raw	10	4	1728	{10080, 10193}
SPOT5	HRG, processed	8	3	1024	1024×3

The corpus includes images from the following instruments: Infrared Atmospheric Sounding Interferometer (IASI), Atmospheric Infrared Sounder (AIRS), Compact Reconnaissance Imaging Spectrometer for Mars (CRISM), Moon Mineralogy Mapper (M3), Hyperion, SWIR Full Spectrum Imager (SFSI), Airborne Visible/Infrared Imaging Spectrometer (AVIRIS), Compact Airborne Spectrographic Imager (CASI), Moderate Resolution Imaging Spectroradiometer (MODIS), Meteosat Second Generation (MSG), Landsat, PLEIADES High Resolution (HR), Vegetation, and Système Pour l'Observation de la Terre 5 (SPOT5) High Resolution Geometric (HRG).

IASI and MSG images are not available for public distribution. All other images can be downloaded at http://cwe.ccsds.org/sls/docs/sls-dc/123.0-B-Info/TestData, both uncompressed and compressed with the Recommended Standard.

For a few of these instruments, more than one type of image is included in the CCSDS test set. The CRISM data includes Full-Resolution Target (FRT) images, as well as Half-Resolution Long (HRL) and MultiSpectral Survey (MSP). The HRL and MSP images are produced onboard the Mars Reconnaissance Orbiter spacecraft by spatial and spectral binning of the samples from the imager. Similarly, M3 'global' images are produced onboard the Chandrayaan-1 spacecraft by spatial and spectral binning of the full resolution 'target' images. The 16-bit AVIRIS images were produced in 2006 from a newer version of the AVIRIS instrument than the 12-bit images which are from 2001 and 2003.

ANNEX B

AVAILABLE SOFTWARE AND TEST PATTERN IMAGE

B1 AVAILABLE SOFTWARE

Software implementations of the Recommended Standard are available via links provided at http://cwe.ccsds.org/sls/docs/sls-dc/123.0-B-Info/Software. These implementations were developed only to demonstrate compression performance; no optimization in speed or programming was guaranteed. The software and data are provided 'as is' to users as a courtesy. In no event will the CCSDS, its member Agencies, or the author of the software be liable for any consequential, incidental, or indirect damages arising out of the use of or inability to use the software.

B2 TEST PATTERN IMAGE

A small test pattern image, along with associated documentation and compressed versions of the image, are available at the following location:

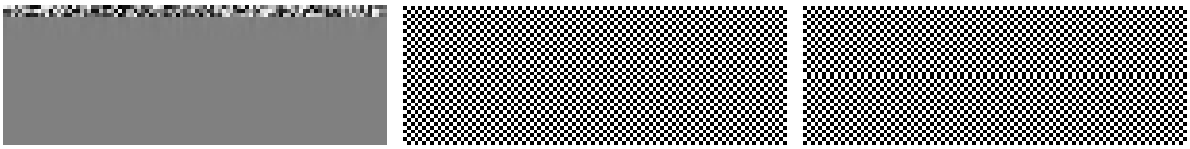
http://cwe.ccsds.org/sls/docs/sls-dc/123.0-B-Info/TestData/TestPattern

This synthetic image is designed to help confirm that an implementation of the CCSDS 123.0-B-1 multispectral and hyperspectral image compressor is compliant with the Recommended Standard. It is a (very unnatural) image designed to exercise several features of the prediction and entropy coding operations specified in the Recommended Standard that might rarely arise for natural images.

The documentation included with the test pattern image indicates compression settings that will:

- cause clipping of weight values to occur (thus helping to confirm correct calculation of equation (34) of reference [1]);
- result in overflow of the register used in the prediction calculation (thus helping to confirm correct calculation of equation (29) of reference [1]);
- cause all allowed values of the variable length code parameter $k_z(t)$ to be used (when sample-adaptive coding is used);
- ensure that the unary length limit ($U_{\max}$) is reached (when sample-adaptive coding is used);
- ensure that the 'zero-block' coding option is used (when block-adaptive coding is used).

Figure B-1 depicts the bands of the test pattern image.



(a) (b) (c)

(a) first band (z=0); (b) bands z=1,2., …, 15; (c) last band (z=16).

Figure B-1: Grayscale Rendering of the Bands of the Test Pattern Image

ANNEX C

COMPRESSION SETTINGS USED FOR EXPERIMENTS

This annex tabulates default compression settings used to derive experimental results in this document.

Default settings for prediction are given in table C-1. Except where noted otherwise, compression results made use of the sample-adaptive entropy coder with settings given in table C-2. In cases where block-adaptive coding was used, the default block size used is J=64.

MODIS is a 'whisk-push' imager that exhibits streaking artifacts parallel to the cross-track direction, unlike typical pushbroom imagers. For this reason, within each segment the y and x dimensions correspond to the cross-track and along-track directions, respectively. This is consistent with Note 1 in subsection 3.2.1 of the Recommended Standard (reference [1]).

Table C-1: Predictor Default Settings for Experimental Results

Name	Symbol	Setting	Description
Number of Prediction Bands	P	3	Number of previous bands used to perform prediction
Register Size	R	64	Size of register used in prediction calculation
Local Sum Type		column-oriented for CRISM, Hyperion raw, M3, MODIS, SFSI rad; neighbor-oriented otherwise	Identifies neighborhood used to calculate local sums
Prediction Mode		reduced for AIRS, CASI raw, CRISM, IASI, M3, MODIS day and night; full otherwise	Indicates whether directional local differences are used in the prediction calculation
Weight Component Resolution	Ω	19	Determines number of bits used to represent each weight vector component
Weight Initialization Method		default	Determines initial values of weight vector components
Weight Initialization Table	$\{\Lambda_z\}$	(unused)	Defines the initial weight components under custom initialization.
Weight Initialization Resolution	Q	(unused)	Determines the precision of the initial weight components under custom initialization.
Weight Update Scaling Exponent Initial Parameter	$v_{\min}$	-1	Determines initial rate at which predictor adapts weight vector to input
Weight Update Scaling Exponent Final Parameter	$v_{\max}$	3	Determines final rate at which predictor adapts weight vector to input
Weight Update Scaling Exponent Change Interval	t_{inc}	2^6	Determines the interval between increments to the weight update scaling exponent

Table C-2: ample-Adaptive Entropy Coder Default Settings for Experimental Results

Name	Symbol	Setting	Description
Unary Length Limit	$U_{\max}$	18	Limits the maximum length of any encoded sample
Rescaling Counter Size	γ^*	6	Determines the interval between rescaling of counter and accumulator
Initial Count Exponent	γ_0	1	Sets initial counter value
Accumulator Initialization Table	$\{k'_z\}$	(unused)	Sets an initial accumulator value for each band
Accumulator Initialization Constant	K	3	Sets initial accumulator value in all bands

ANNEX D

SELECTION CRITERIA

D1 INTRODUCTION

In October 2007, the MHDC working group established selection criteria for the evaluation of compressors proposed for standardization. These criteria were intended to encompass both lossless and lossy compressors, and consequently some of the selection criteria are not applicable to lossless compressors. The text of the selection criteria document forms the remainder of this annex, with references renumbered to match the reference numbers used in 1.6 of this document.

In addition to the mandatory requirements, the following selection criteria will be considered by the MHDC working group in the evaluation of proposed compressors:

a) implementation complexity;

b) compression effectiveness;

c) flexibility;

d) error containment;

e) robustness;

f) user-friendliness;

g) license considerations.

Each of these criteria is described in further detail below.

D2 IMPLEMENTATION COMPLEXITY

The implementation complexity considerations are:

a) throughput of hardware implementation;

b) algorithmic complexity (number of arithmetic operations per sample or other relevant metric);

c) memory/buffering requirements.

D3 COMPRESSION EFFECTIVENESS

D3.1 GENERAL

Compression effectiveness refers to the ability of a compressor to provide high-fidelity (or lossless) reconstructed data using a small compressed data volume, without regard to other concerns such as speed or memory use.

For compressors offering flexibility in adjusting compression parameters, there should be provided compression-effectiveness results that indicate performance in a variety of configurations, e.g., scan-based vs. frame-based, small vs. large error-containment segment size, default vs. custom decorrelating transform, etc.

D3.2 LOSSLESS COMPRESSION

For lossless compression, compression effectiveness is measured by the bit rate required to achieve lossless compression on each test data set.

D3.3 LOSSY COMPRESSION

Lossy compression effectiveness is measured by the rate-distortion performance on the test data sets. Unless indicated otherwise in the README file associated with a test data set, lossy compression algorithms should provide distortion results:

a) at or near compressed bit rates of 4, 2, 1, 0.5, 0.25, 0.1 bits/sample (here a ‘sample’ refers to a single integer output value from a detector element); and

b) at or near compression ratios of 10:1 and 20:1.

At the bit rates examined, the following distortion metrics (described in the references) will be considered:

– RRMSE (reference [40]);

– F_λ (reference [40]);

– Q(x,y) (reference [40]);

– MAE (reference [40]);

– MAD (reference [40]);

– $\mathrm{SNR} = 10\log_{10}\left(\dfrac{\text{signal power}}{\text{MSE}}\right)$ (reference [41]);

– ‘classical’ PSNR:

$$\mathrm{PSNR} = 20\log_{10}\left(\frac{2^b - 1}{\sqrt{\frac{1}{N_x N_y N_\lambda}\sum_{x,y,\lambda}(s_{x,y,\lambda} - \hat{s}_{x,y,\lambda})^2}}\right) \text{(dB)};$$

– ‘compression noise’ (reference [42]) (RMS error in each band, normalized by the standard deviation of the band).

In addition, as a check for bias in reconstructed spectral bands, the following bias metric should be evaluated:

$$b = \max_{\lambda}\frac{1}{N_x N_y}\left|\sum_{x,y} s_{x,y,\lambda} - \hat{s}_{x,y,\lambda}\right|$$

Here $s_{x,y,\lambda}$ denotes the sample value at spatial position (x,y) in spectral band λ; $\hat{s}_{x,y,\lambda}$ denotes the reconstructed value of $s_{x,y,\lambda}$; N_x, N_y, and N_λ denote the number of columns, rows and spectral bands in the data set; b denotes the instrument bit depth.

D4 FLEXIBILITY

Desirable flexibility in the compressor includes features such as:

a) flexibility in the ability to control the tradeoff between rate and distortion (e.g., bit-accurate rate control, fine granularity of quality levels, and the ability to specify rate and/or quality);

b) the ability to provide both lossless and lossy compression (for lossy compressors, the ability to provide effective compression over a wide range of bit rates or fidelity is desirable);

c) the ability to tune the algorithm post-launch to improve performance (e.g., uploading new compression parameters to improve performance if, for example, a detector element was bad);

d) flexibility in the structure of the compressed bitstream (e.g., progressive transmission, the ability to provide higher reconstructed fidelity or downlink priority in certain spectral bands or regions-of-interest, etc.).

D5 ERROR CONTAINMENT

The effectiveness of the error-containment mechanism to deal with packet losses on the communications channel will be considered. Considerations include the ability to adjust the size of error-containment segments and the impact of using small error-containment segments on compression effectiveness. In certain applications, error concealment following data loss or corruption may be desirable.

D6 USER-FRIENDLINESS

A desirable feature of a compressor is that the algorithm achieves good performance without significant expertise on the part of the user implementing the algorithm. Ideally, the algorithm should be operated as a black box; e.g., no tuning of coding tables would be required from user.

D7 ROBUSTNESS

Instrument defects tend to adversely affect compression performance. It is desirable for compression effectiveness to not be dramatically affected by

a) detector defects (dead pixels, frozen pixels, spikes, etc.); or

b) instrument defects such as spectral and spatial misregistration, keystone, and smile.

D8 LICENSE CONSIDERATIONS

In addition to the mandatory requirement that CCSDS member agencies may use a recommended compressor without payment of a license fee, other license considerations may be relevant. For example, the ability of a 3rd party to use a recommended algorithm for a commercial venture without paying a license fee may be desirable because it provides further incentive for such a party to develop a hardware implementation.

ANNEX E

COMPRESSION RESULTS

Compressed data rates, measured in bits/sample, are given in tables E-1 and E-2 for full image compression of hyperspectral and multispectral images, respectively. Results for segmented images are given in tables E-3 and E-4 for hyperspectral and multispectral images, respectively.

Table E-1: Compressed Data Rates (in Bits/Sample) for Full Image Compression of Hyperspectral Images

		CCSDS-123.0-B										JPEG-LS		ESA + sample-adaptive coding	LUT + sample-adaptive coding	CCSDS-122.0-B			JPEG2000		
		defaults	sample-adaptive			block-adaptive BIL			block-adaptive BIP												
Instrument	scene		reduced+ column	full+ neighbor	reduced+ neighbor	reduced+ column	full+ neighbor	reduced+ neighbor	reduced+ column	full+ neighbor	reduced+ neighbor	direct	differential			direct	+ IWT	+ POT	direct	+ IWT	+ POT
IASI	desert	4.70	5.03	4.74	4.70	5.07	4.78	4.75	5.02	4.73	4.70	6.25	4.95	4.95	5.66	7.12	5.53	6.1	6.62	5.12	5.58
IASI	ecosse	4.91	5.27	5.02	4.91	5.31	5.07	4.95	5.19	4.97	4.86	7.29	5.17	4.96	5.55	8.37	5.83	6.19	7.76	5.41	5.61
IASI	polenord	4.64	4.97	4.70	4.64	5.01	4.73	4.68	4.91	4.65	4.60	6.34	4.86	4.70	5.17	7.21	5.43	5.92	6.73	5.03	5.34
IASI	tropocean	4.77	5.09	4.80	4.77	5.13	4.85	4.81	5.10	4.83	4.79	6.53	5.10	5.04	5.89	7.5	5.72	6.24	6.95	5.28	5.68
IASI	***average***	***4.75***	***5.09***	***4.82***	***4.75***	***5.13***	***4.86***	***4.80***	***5.06***	***4.79***	***4.74***	***6.60***	***5.02***	***4.91***	***5.57***	***7.55***	***5.63***	***6.11***	***7.02***	***5.21***	***5.55***
AIRS	9	4.22	4.54	4.24	4.22	4.60	4.29	4.28	4.64	4.32	4.32	6.35	4.77	4.59	5.53	6.91	5.01	4.94	6.69	4.76	4.58
AIRS	16	4.20	4.50	4.20	4.20	4.56	4.25	4.25	4.59	4.29	4.29	6.21	4.68	4.53	5.41	6.67	4.87	4.9	6.44	4.6	4.52
AIRS	60	4.38	4.71	4.40	4.38	4.77	4.46	4.44	4.81	4.50	4.49	6.84	4.98	4.81	5.85	7.38	5.25	5.16	7.15	5	4.8
AIRS	82	4.13	4.44	4.13	4.13	4.49	4.18	4.18	4.52	4.21	4.21	5.86	4.56	4.38	5.21	6.27	4.71	4.79	6.04	4.43	4.4
AIRS	120	4.29	4.62	4.29	4.29	4.68	4.35	4.34	4.72	4.38	4.38	6.32	4.80	4.66	5.49	6.84	5.02	5.01	6.61	4.77	4.65
AIRS	126	4.40	4.73	4.42	4.40	4.78	4.48	4.46	4.82	4.51	4.50	6.70	4.99	4.79	5.82	7.26	5.25	5.16	7.01	4.99	4.81
AIRS	129	4.14	4.43	4.13	4.14	4.49	4.19	4.19	4.52	4.21	4.22	5.56	4.53	4.45	5.32	6.03	4.67	4.8	5.78	4.39	4.43
AIRS	151	4.41	4.73	4.43	4.41	4.79	4.49	4.47	4.81	4.50	4.49	6.45	4.95	4.81	5.90	6.94	5.15	5.14	6.74	4.91	4.8
AIRS	182	4.43	4.72	4.46	4.43	4.78	4.51	4.49	4.80	4.53	4.52	6.51	4.96	4.90	6.26	7.07	5.22	5.21	6.82	4.96	4.85
AIRS	193	4.42	4.75	4.44	4.42	4.81	4.49	4.47	4.85	4.53	4.52	6.65	4.99	4.84	5.75	7.18	5.24	5.17	6.96	5	4.83
AIRS	***average***	***4.30***	***4.62***	***4.31***	***4.30***	***4.67***	***4.37***	***4.36***	***4.71***	***4.40***	***4.39***	***6.35***	***4.82***	***4.68***	***5.66***	***6.86***	***5.04***	***5.03***	***6.62***	***4.78***	***4.67***
CRISM-FRT	frt00006d1c_07_sc164	4.61	4.61	5.57	8.41	4.66	5.57	8.40	4.66	5.57	8.52	4.67	5.12	8.21	8.99	6.18	6.44	6.48	5.03	5.31	5.46
CRISM-FRT	frt00010f86_07_sc167	5.28	5.28	6.90	9.48	5.33	6.90	9.49	5.32	6.99	9.71	5.29	5.80	9.46	9.96	6.98	7.39	7.43	5.85	6.27	6.48
CRISM-FRT	frt00013e49_07_sc166	5.46	5.46	7.27	9.52	5.52	7.28	9.53	5.49	7.35	9.76	6.07	5.86	9.44	10.02	7.48	7.45	7.52	6.6	6.39	6.67
CRISM-FRT	frt000065e6_07_sc164	4.83	4.83	5.84	8.46	4.88	5.84	8.46	4.88	5.81	8.57	5.14	5.26	8.37	9.14	6.5	6.53	6.56	5.5	5.45	5.57
CRISM-FRT	frt0000869b_07_sc164	5.06	5.06	6.75	8.79	5.10	6.75	8.80	5.08	6.70	8.93	6.12	5.42	8.73	9.44	7.21	6.86	6.89	6.48	5.81	5.99
CRISM-FRT	frt0001077d_07_sc166	5.39	5.39	6.97	9.45	5.45	6.98	9.46	5.42	7.05	9.68	5.61	5.82	9.36	9.95	7.14	7.37	7.4	6.14	6.3	6.52
CRISM-FRT	frt0001241d_07_sc164	5.35	5.35	6.83	9.27	5.40	6.84	9.28	5.38	6.87	9.46	5.70	5.76	9.19	9.85	7.18	7.3	7.32	6.22	6.22	6.42
CRISM-FRT	frt00008849_07_sc165	4.88	4.88	6.22	8.76	4.94	6.22	8.76	4.92	6.19	8.90	5.31	5.29	9.19	9.39	6.73	6.75	6.79	5.74	5.62	5.81
CRISM-FRT	frt00009326_07_sc167	4.65	4.65	6.54	8.52	4.70	6.55	8.51	4.66	6.49	8.65	5.83	4.99	8.35	9.09	6.9	6.48	6.64	6.19	5.4	5.78
CRISM-FRT	***average (FRT)***	***5.06***	***5.06***	***6.54***	***8.96***	***5.11***	***6.55***	***8.97***	***5.09***	***6.56***	***9.13***	***5.53***	***5.48***	***8.92***	***9.54***	***6.92***	***6.95***	***7.00***	***5.97***	***5.86***	***6.08***
CRISM-HRL	hrl0000ba9c_07_sc183	4.92	4.92	6.20	8.96	4.97	6.21	8.97	4.96	6.25	9.17	5.29	5.37	8.91	9.47	6.79	6.87	6.95	5.81	5.8	6
CRISM-HRL	hrl00004f38_07_sc181	4.26	4.26	6.22	7.99	4.30	6.22	7.98	4.26	6.15	8.10	6.08	4.59	7.90	8.66	6.98	6.01	6.18	6.34	4.94	5.32
CRISM-HRL	hrl0000891d_07_sc182	4.58	4.58	5.70	8.54	4.63	5.72	8.55	4.61	5.71	8.69	5.03	4.99	8.49	9.15	6.53	6.48	6.56	5.5	5.35	5.53
CRISM-HRL	hrl00006548f_07_sc182	4.46	4.46	6.15	8.28	4.50	6.15	8.27	4.47	6.10	8.41	5.82	4.81	8.22	8.95	6.89	6.28	6.4	6.15	5.19	5.46
CRISM-HRL	***average (HRL)***	***4.56***	***4.56***	***6.07***	***8.44***	***4.60***	***6.08***	***8.44***	***4.57***	***6.05***	***8.59***	***5.55***	***4.94***	***8.38***	***9.06***	***6.80***	***6.41***	***6.52***	***5.95***	***5.32***	***5.58***
CRISM-MSP	msp00003e34_01_sc214	2.37	2.37	3.12	6.51	2.41	3.15	6.54	2.46	3.19	6.95	2.84	2.73	6.68	6.94	4.53	4.53	5.06	3.33	3.29	3.79
CRISM-MSP	msp00003ea5_07_sc214	2.37	2.37	3.38	6.48	2.41	3.42	6.51	2.46	3.44	6.93	3.19	2.77	6.63	6.94	4.68	4.56	5.16	3.61	3.35	4
CRISM-MSP	msp00003fbc_03_sc214	2.42	2.42	3.73	6.41	2.46	3.77	6.45	2.52	3.75	6.84	3.68	2.86	6.60	6.93	4.97	4.58	5.22	4.03	3.41	4.04
CRISM-MSP	msp0000426b_01_sc214	2.33	2.33	2.86	6.48	2.37	2.89	6.51	2.42	2.95	6.90	2.55	2.73	6.68	6.74	4.41	4.5	4.93	3.04	3.22	3.46
CRISM-MSP	msp0000438d_03_sc214	2.36	2.36	3.47	6.42	2.41	3.51	6.45	2.47	3.50	6.86	3.36	2.77	6.58	6.87	4.78	4.53	5.21	3.74	3.33	3.95
CRISM-MSP	msp0001081c_07_sc214	3.38	3.38	4.18	7.33	3.39	4.19	7.35	3.57	4.42	8.12	3.75	3.84	7.61	7.61	5.39	5.53	5.94	4.25	4.38	4.74
CRISM-MSP	msp00004125_05_sc214	2.62	2.62	4.55	6.51	2.66	4.59	6.54	2.71	4.59	6.92	4.60	3.21	6.79	7.14	5.64	4.84	5.24	4.96	3.8	4.15
CRISM-MSP	***average (MSP)***	***2.55***	***2.55***	***3.61***	***6.59***	***2.59***	***3.65***	***6.62***	***2.66***	***3.69***	***7.08***	***3.42***	***2.99***	***6.80***	***7.02***	***4.91***	***4.72***	***5.25***	***3.85***	***3.54***	***4.02***
M3-Global	globalA	2.17	2.17	6.10	6.99	2.22	6.13	6.93	2.19	5.16	6.71	4.89	2.59	6.49	7.22	5.75	4.31	4.52	5.14	3.15	3.61
M3-Global	globalB	2.12	2.12	6.08	6.97	2.17	6.12	6.91	2.15	5.08	6.68	4.77	2.54	6.41	7.18	5.69	4.26	4.48	5.05	3.1	3.5
M3-Global	***average (global)***	***2.14***	***2.14***	***6.09***	***6.98***	***2.19***	***6.12***	***6.92***	***2.17***	***5.12***	***6.69***	***4.83***	***2.56***	***6.45***	***7.20***	***5.72***	***4.29***	***4.50***	***5.10***	***3.13***	***3.56***
M3-Target	targetA	3.38	3.38	5.18	6.98	3.40	5.13	6.96	3.39	4.39	6.87	3.83	3.58	6.88	7.68	5.24	4.84	4.94	4.19	3.75	3.87
M3-Target	targetB	3.18	3.18	5.09	6.68	3.21	5.06	6.65	3.21	4.35	6.59	3.95	3.39	6.70	7.59	5.13	4.64	4.72	4.21	3.56	3.69
M3-Target	targetC	2.71	2.71	4.97	6.47	2.74	4.89	6.43	2.75	3.96	6.30	3.20	3.01	6.22	7.04	4.7	4.33	4.43	3.59	3.2	3.36
M3-Target	***average (target)***	***3.09***	***3.09***	***5.08***	***6.71***	***3.12***	***5.03***	***6.68***	***3.12***	***4.24***	***6.59***	***3.66***	***3.33***	***6.60***	***7.44***	***5.02***	***4.60***	***4.70***	***4.00***	***3.50***	***3.64***
Hyperion	ErtaAle	4.26	4.29	4.60	5.20	4.34	4.64	5.24	4.42	4.72	5.44	4.97	4.54	5.47	6.06	5.54	5.02	5.02	5.07	4.47	4.5
Hyperion	LakeMonona	4.39	4.42	4.69	5.41	4.47	4.73	5.45	4.57	4.80	5.66	4.95	4.67	5.73	6.32	5.54	5.14	5.18	5.08	4.59	4.69
Hyperion	MtStHelens	4.30	4.33	4.72	5.24	4.38	4.76	5.27	4.48	4.89	5.49	5.18	4.60	5.52	6.06	5.71	5.11	5.12	5.31	4.59	4.65
Hyperion	Cuprite	4.26	4.29	4.57	5.07	4.34	4.61	5.11	4.42	4.68	5.31	4.99	4.53	5.35	5.89	5.54	4.94	4.96	5.1	4.42	4.49
Hyperion	***average (raw)***	***4.30***	***4.33***	***4.65***	***5.23***	***4.38***	***4.69***	***5.27***	***4.47***	***4.77***	***5.47***	***5.02***	***4.59***	***5.52***	***6.08***	***5.58***	***5.05***	***5.07***	***5.14***	***4.52***	***4.58***
Hyperion	Cuprite_flatfield	3.97	4.29	3.97	4.00	4.34	4.01	4.04	4.42	4.07	4.11	4.80	4.33	4.11	4.54	4.99	4.39	4.32	4.89	4.19	4.05
SFSI	mantar_raw	4.67	4.85	4.67	4.76	4.89	4.70	4.80	5.04	4.71	4.93	4.75	5.02	4.91	5.43	5.3	4.97	4.9	4.65	4.61	4.56
SFSI	mantar_rad_rmnoise	2.96	2.97	3.71	3.82	2.86	3.65	3.75	3.55	4.56	4.75	4.35	3.08	4.07	4.70	4.88	3.92	3.75	4.42	3.34	3.23
AVIRIS	sc0_raw	6.19	6.54	6.19	6.20	6.58	6.23	6.24	6.81	6.45	6.47	9.16	6.95	6.45	7.15	9.31	7.13	6.7	9.46	7.13	6.65
AVIRIS	sc3_raw	6.06	6.40	6.06	6.06	6.43	6.10	6.10	6.66	6.32	6.33	8.85	6.83	6.30	6.93	9.06	7.01	6.54	9.19	7	6.47
AVIRIS	sc10_raw	5.58	5.92	5.58	5.59	5.95	5.60	5.62	6.18	5.82	5.84	7.29	6.16	5.62	6.23	7.46	6.18	5.93	7.45	6.09	5.8
AVIRIS	sc11_raw	5.83	6.17	5.83	5.84	6.21	5.87	5.88	6.42	6.07	6.09	8.48	6.48	6.03	6.80	8.64	6.65	6.36	8.74	6.6	6.27
AVIRIS	sc18_raw	6.21	6.53	6.21	6.19	6.56	6.23	6.21	6.79	6.47	6.46	9.28	6.94	6.39	7.22	9.42	7.12	6.76	9.58	7.12	6.71
AVIRIS	***average (16-bit raw)***	***5.98***	***6.31***	***5.98***	***5.98***	***6.35***	***6.01***	***6.01***	***6.57***	***6.22***	***6.24***	***8.61***	***6.67***	***6.16***	***6.87***	***8.78***	***6.82***	***6.46***	***8.88***	***6.79***	***6.38***
AVIRIS	hawaii_sc01_raw	2.62	2.94	2.62	2.63	2.99	2.68	2.69	3.06	2.76	2.77	4.58	3.27	2.94	3.38	4.77	3.51	3.29	4.62	3.26	2.99
AVIRIS	maine_sc10_raw	2.73	3.03	2.73	2.73	3.08	2.77	2.78	3.16	2.86	2.87	4.50	3.36	3.07	3.51	4.66	3.62	3.36	4.55	3.41	3.09
AVIRIS	***average (12-bit raw)***	***2.68***	***2.99***	***2.68***	***2.68***	***3.04***	***2.72***	***2.73***	***3.11***	***2.81***	***2.82***	***4.54***	***3.32***	***3.01***	***3.44***	***4.72***	***3.57***	***3.33***	***4.59***	***3.34***	***3.04***
AVIRIS	sc0_cal	3.96	4.29	3.96	3.97	4.33	4.00	4.01	4.59	4.24	4.27	6.92	4.73	4.59	4.85	7.1	4.86	4.63	7.14	4.73	4.46
AVIRIS	sc3_cal	3.83	4.15	3.83	3.84	4.19	3.87	3.88	4.45	4.11	4.14	6.65	4.63	4.46	4.65	6.85	4.75	4.5	6.86	4.62	4.31
AVIRIS	sc10_cal	3.37	3.71	3.37	3.39	3.74	3.41	3.42	4.04	3.70	3.72	5.16	4.01	3.80	3.93	5.29	4.04	3.94	5.15	3.83	3.68
AVIRIS	sc11_cal	3.64	3.98	3.64	3.65	4.02	3.68	3.70	4.28	3.93	3.95	6.21	4.28	4.18	4.43	6.42	4.41	4.31	6.39	4.24	4.1
AVIRIS	sc18_cal	3.91	4.24	3.91	3.91	4.28	3.95	3.95	4.57	4.23	4.25	6.99	4.68	4.56	4.86	7.2	4.82	4.68	7.24	4.69	4.51
AVIRIS	***average (16-bit cal)***	***3.74***	***4.07***	***3.74***	***3.75***	***4.11***	***3.78***	***3.79***	***4.39***	***4.04***	***4.06***	***6.39***	***4.46***	***4.32***	***4.54***	***6.57***	***4.58***	***4.41***	***6.56***	***4.42***	***4.21***
CASI	t0180f07_raw	4.86	5.16	4.88	4.86	5.19	4.91	4.89	5.29	5.02	5.00	6.48	5.23	4.96	5.49	6.66	5.3	5.37	6.6	5.11	5.16
CASI	t0477f06_raw	5.18	5.40	5.27	5.18	5.43	5.30	5.22	5.51	5.39	5.28	7.07	5.44	5.23	5.82	7.38	5.61	5.74	7.34	5.44	5.57
CASI	***average (raw)***	***5.02***	***5.28***	***5.07***	***5.02***	***5.31***	***5.10***	***5.05***	***5.40***	***5.20***	***5.14***	***6.77***	***5.33***	***5.10***	***5.65***	***7.02***	***5.46***	***5.56***	***6.97***	***5.28***	***5.37***
CASI	t0180f07_rad	7.58	7.96	7.58	7.61	8.00	7.62	7.66	8.24	7.83	7.89	9.23	7.91	7.79	8.33	9.41	7.96	8.06	9.46	7.87	7.98
CASI	t0477f06_rad_rss	8.30	8.62	8.30	8.29	8.66	8.33	8.33	8.79	8.45	8.48	10.35	8.56	8.56	9.19	10.59	8.73	8.81	10.7	8.7	8.8

Table E-2: Compressed Data Rates (in Bits/Sample) for Full Image Compression of Multispectral Images

		CCSDS-123.0-B										JPEG-LS		ESA + sample-adaptive coding	LUT + sample-adaptive coding	CCSDS-122.0-B			JPEG2000		
		defaults	sample-adaptive			block-adaptive BIL			block-adaptive BIP												
Instrument	scene		reduced+ column	full+ neighbor	reduced+ neighbor	reduced+ column	full+ neighbor	reduced+ neighbor	reduced+ column	full+ neighbor	reduced+ neighbor	direct	differential			direct	+ IWT	+ POT	direct	+ IWT	+ POT
MODIS-night	A2001123.0000_night	4.44	4.44	4.96	7.01	4.47	4.98	7.05	5.00	5.29	7.31	4.95	5.36	7.69	6.93	5.89	6.04	4.73	5.13	5.25	4.53
MODIS-night	A2001123.1630_night	5.10	5.10	5.74	7.24	5.12	5.76	7.28	5.97	6.34	7.54	5.97	6.43	7.88	7.65	6.61	6.85	5.63	6.09	6.36	5.52
MODIS-night	A2001222.0835_night	4.45	4.45	5.00	7.09	4.47	5.02	7.12	5.02	5.37	7.43	4.99	5.43	7.63	6.94	5.91	6.13	4.83	5.1	5.34	4.62
MODIS-night	A2001222.0840_night	5.15	5.15	5.80	7.34	5.18	5.84	7.39	6.16	6.48	7.70	6.09	6.58	7.96	7.71	6.76	7.05	5.71	6.23	6.59	5.6
MODIS-night	A2001222.1200_night	4.37	4.37	4.86	7.05	4.38	4.87	7.09	5.06	5.31	7.43	4.89	5.41	7.62	6.90	5.84	6.13	4.85	5.02	5.34	4.64
MODIS-night	***average (night)***	***4.70***	***4.70***	***5.27***	***7.14***	***4.72***	***5.29***	***7.18***	***5.44***	***5.76***	***7.48***	***5.38***	***5.84***	***7.76***	***7.22***	***6.20***	***6.44***	***5.15***	***5.51***	***5.78***	***4.98***
MODIS-day	A2001123.0000_day	4.21	4.21	4.18	4.26	3.53	3.50	3.57	6.48	6.21	6.52	2.83	3.07	4.08	4.56	3.51	5.01	4.67	3.31	4.89	4.29
MODIS-day	A2001123.1630_day	6.04	6.04	6.05	6.10	5.73	5.76	5.81	7.34	7.36	7.50	5.07	5.58	6.02	6.94	5.91	7.13	7.15	5.74	7.02	6.99
MODIS-day	A2001222.0835_day	6.37	6.37	6.58	6.59	5.97	6.24	6.26	6.96	7.17	7.28	5.38	5.66	6.48	7.31	6.44	7.07	6.99	6.15	6.88	6.77
MODIS-day	A2001222.0840_day	6.22	6.22	6.19	6.21	5.98	5.94	5.96	7.62	7.53	7.63	5.13	5.67	6.26	6.66	6.18	7.42	7.17	5.96	7.33	6.96
MODIS-day	A2001222.1200_day	5.78	5.78	5.94	5.94	5.34	5.55	5.57	6.69	6.89	7.02	5.03	5.18	5.91	6.72	5.9	6.61	6.64	5.68	6.4	6.38
MODIS-day	***average (day)***	***5.72***	***5.72***	***5.79***	***5.82***	***5.31***	***5.40***	***5.43***	***7.02***	***7.03***	***7.19***	***4.69***	***5.03***	***5.75***	***6.44***	***5.59***	***6.65***	***6.52***	***5.37***	***6.50***	***6.28***
MODIS-500m	A2001123.0000_500m	6.82	7.42	7.18	7.78	7.46	7.21	7.82	8.57	7.42	8.38	7.56	6.37	8.38	8.96	8.12	7.88	7.19	7.73	6.78	6.58
MODIS-500m	A2001123.1630_500m	7.52	7.98	7.70	8.09	7.98	7.71	8.11	8.93	8.02	8.71	7.94	7.06	8.48	9.11	8.42	8.37	7.85	8.09	7.61	7.41
MODIS-500m	A2001222.0835_500m	6.92	7.55	7.00	7.57	7.56	7.01	7.59	8.84	7.47	8.54	7.17	6.25	8.11	8.51	7.89	7.97	7.33	7.25	6.8	6.54
MODIS-500m	A2001222.0840_500m	7.68	8.21	7.68	8.14	8.23	7.69	8.16	9.16	8.13	8.96	8.06	7.23	8.56	9.14	8.54	8.57	8.06	8.27	7.94	7.65
MODIS-500m	A2001222.1200_500m	7.04	7.61	7.17	7.75	7.62	7.17	7.76	8.92	7.56	8.70	7.39	6.28	8.28	8.66	8.11	8.02	7.39	7.53	6.78	6.58
MODIS-500m	***average (500m)***	***7.20***	***7.75***	***7.35***	***7.87***	***7.77***	***7.36***	***7.89***	***8.88***	***7.72***	***8.66***	***7.62***	***6.64***	***8.36***	***8.88***	***8.22***	***8.16***	***7.56***	***7.77***	***7.18***	***6.95***
MODIS-250m	A2001123.0000_250m	5.83	6.11	6.07	6.32	6.12	6.08	6.34	6.53	6.23	6.57	6.88	6.29	6.79	6.84	6.95	6.59	5.64	6.94	6.52	5.41
MODIS-250m	A2001123.1630_250m	7.11	7.38	7.10	7.29	7.33	7.06	7.25	7.48	7.18	7.39	7.19	6.99	7.44	8.35	7.36	7.37	7.2	7.34	7.33	7.16
MODIS-250m	A2001222.0835_250m	6.16	6.35	6.47	6.79	6.32	6.47	6.80	6.58	6.66	6.92	6.65	6.35	6.86	7.51	6.84	6.86	6.25	6.7	6.71	6.15
MODIS-250m	A2001222.0840_250m	7.33	7.63	7.35	7.61	7.61	7.35	7.61	7.84	7.54	7.79	7.45	7.22	7.90	8.54	7.68	7.84	7.52	7.69	7.87	7.56
MODIS-250m	A2001222.1200_250m	6.00	6.18	6.45	6.76	6.13	6.44	6.76	6.39	6.54	6.86	6.61	6.25	6.78	7.40	6.87	6.71	5.9	6.75	6.54	5.75
MODIS-250m	***average (250m)***	***6.48***	***6.73***	***6.69***	***6.95***	***6.70***	***6.68***	***6.95***	***6.96***	***6.83***	***7.11***	***6.96***	***6.62***	***7.15***	***7.73***	***7.14***	***7.07***	***6.50***	***7.08***	***6.99***	***6.41***
MSG	3	3.34	3.78	3.34	3.54	3.74	3.30	3.49	4.06	3.58	3.81	3.43	3.80	4.06	5.01	3.62	4.04	3.86	3.43	3.86	3.67
MSG	15	3.54	4.02	3.54	3.74	3.99	3.51	3.70	4.29	3.77	4.00	3.69	3.96	4.26	5.25	3.89	4.17	4.02	3.69	3.96	3.81
MSG	31	3.30	3.75	3.30	3.52	3.71	3.27	3.47	3.99	3.51	3.73	3.37	3.73	4.01	4.93	3.58	3.97	3.81	3.38	3.78	3.61
MSG	***average***	***3.39***	***3.85***	***3.39***	***3.60***	***3.81***	***3.36***	***3.56***	***4.11***	***3.62***	***3.85***	***3.50***	***3.83***	***4.11***	***5.06***	***3.70***	***4.06***	***3.90***	***3.50***	***3.87***	***3.70***
LANDSAT	agriculture	3.60	3.96	3.60	3.71	3.97	3.62	3.73	3.99	3.65	3.77	3.99	3.81	4.51	4.82	4.38	4.07	3.99	4.27	3.92	3.82
LANDSAT	coast	2.74	3.06	2.74	2.76	3.06	2.74	2.76	3.11	2.76	2.79	2.93	2.97	2.89	3.50	3.25	3.16	3.18	2.99	2.86	2.88
LANDSAT	mountain	3.78	4.15	3.78	3.84	4.17	3.80	3.87	4.39	3.99	4.11	4.16	4.08	4.32	4.65	4.45	4.37	4.25	4.34	4.22	4.11
LANDSAT	***average***	***3.37***	***3.73***	***3.37***	***3.44***	***3.74***	***3.39***	***3.45***	***3.83***	***3.47***	***3.56***	***3.70***	***3.62***	***3.91***	***4.32***	***4.03***	***3.87***	***3.81***	***3.87***	***3.67***	***3.60***
PLEIADES	montp_trans	7.42	7.77	7.42	7.56	7.83	7.48	7.62	8.15	7.74	7.95	7.98	7.40	8.05	8.70	8.2	7.76	8.02	8.3	7.84	8.11
PLEIADES	perp_trans	7.21	7.52	7.21	7.36	7.53	7.24	7.39	7.70	7.37	7.57	7.69	7.22	7.85	8.61	7.89	7.4	7.64	8	7.47	7.72
PLEIADES	***average***	***7.32***	***7.64***	***7.32***	***7.46***	***7.68***	***7.36***	***7.50***	***7.93***	***7.55***	***7.76***	***7.84***	***7.31***	***7.95***	***8.65***	***8.05***	***7.58***	***7.83***	***8.15***	***7.66***	***7.92***
PLEIADES	montp_misreg0	6.71	7.03	6.71	6.84	7.07	6.75	6.88	7.47	7.11	7.29	7.24	6.67	7.31	7.99	7.46	7.1	7.29	7.53	7.15	7.33
PLEIADES	montp_misreg20	6.98	7.27	6.98	7.12	7.31	7.01	7.15	7.64	7.29	7.48	7.24	6.88	7.53	8.16	7.46	7.26	7.48	7.53	7.31	7.54
PLEIADES	montp_misreg35	7.14	7.44	7.14	7.29	7.46	7.17	7.32	7.74	7.40	7.58	7.24	7.05	7.69	8.31	7.46	7.37	7.59	7.53	7.42	7.66
PLEIADES	montp_misreg50	7.26	7.57	7.26	7.42	7.59	7.28	7.45	7.82	7.47	7.66	7.24	7.19	7.83	8.44	7.46	7.44	7.66	7.53	7.49	7.74
VEGETATION	1_1b	5.24	5.28	5.24	5.72	5.19	5.17	5.62	5.32	5.30	5.99	5.29	5.04	5.82	6.93	5.56	5.52	5.5	5.46	5.36	5.33
VEGETATION	2_1b	5.28	5.31	5.28	5.76	5.23	5.21	5.66	5.37	5.34	6.03	5.32	5.06	5.90	7.00	5.61	5.56	5.56	5.5	5.41	5.39
VEGETATION	***average (level 1b)***	***5.26***	***5.30***	***5.26***	***5.74***	***5.21***	***5.19***	***5.64***	***5.35***	***5.32***	***6.01***	***5.30***	***5.05***	***5.86***	***6.96***	***5.59***	***5.54***	***5.53***	***5.48***	***5.39***	***5.36***
VEGETATION	1_1c	5.02	5.29	5.02	5.14	5.21	4.95	5.06	5.34	5.03	5.18	5.25	5.01	5.51	6.73	5.44	5.32	5.26	5.4	5.28	5.21
VEGETATION	2_1c	5.06	5.33	5.06	5.18	5.25	4.99	5.10	5.38	5.08	5.23	5.28	5.03	5.58	6.80	5.48	5.37	5.33	5.44	5.33	5.28
VEGETATION	***average (level 1c)***	***5.04***	***5.31***	***5.04***	***5.16***	***5.23***	***4.97***	***5.08***	***5.36***	***5.05***	***5.20***	***5.26***	***5.02***	***5.55***	***6.77***	***5.46***	***5.35***	***5.30***	***5.42***	***5.31***	***5.25***
SPOT5	1	5.14	5.48	5.14	5.25	5.48	5.14	5.25	5.50	5.17	5.30	5.44	5.39	5.63	6.04	5.64	5.7	5.34	5.61	5.68	5.28
SPOT5	2	4.27	4.64	4.27	4.37	4.63	4.26	4.35	4.65	4.28	4.39	4.35	4.39	4.89	5.43	4.63	4.79	4.4	4.58	4.74	4.3
SPOT5	3	4.17	4.60	4.17	4.34	4.60	4.17	4.34	4.60	4.16	4.35	4.32	4.36	4.99	5.52	4.61	4.73	4.36	4.57	4.68	4.26
SPOT5	***average***	***4.53***	***4.91***	***4.53***	***4.65***	***4.90***	***4.52***	***4.65***	***4.92***	***4.54***	***4.68***	***4.71***	***4.71***	***5.17***	***5.66***	***4.96***	***5.07***	***4.70***	***4.92***	***5.03***	***4.61***

Table E-3: Compressed Data Rates (in Bits/Sample) for Segmented Compression of Hyperspectral Images

			sample-adaptive			block-adaptive BIL			block-adaptive BIP					ESA + sample-adaptive coding	LUT + sample-adaptive coding						
Instrument	scene	defaults	reduced+ column	full+ neighbor	reduced+ neighbor	reduced+ column	full+ neighbor	reduced+ neighbor	reduced+ column	full+ neighbor	reduced+ neighbor	direct	differential			direct	+ IWT	+ POT	direct	+ IWT	+ POT
IASI	desert	4.72	5.04	4.76	4.72	5.08	4.80	4.76	5.03	4.75	4.71	6.25	4.96	4.95	5.83	7.19	5.6	6.17	**7.01**	**5.54**	**6.02**
IASI	ecosse	4.92	5.27	5.04	4.92	5.31	5.08	4.96	5.20	4.98	4.87	7.29	5.19	4.96	5.68	8.45	5.9	6.25	**8.17**	**5.84**	**6.08**
IASI	polenord	4.65	4.98	4.72	4.65	5.01	4.75	4.69	4.92	4.67	4.61	6.34	4.87	4.70	5.23	7.28	5.49	5.98	**7.14**	**5.46**	**5.81**
IASI	tropocean	4.78	5.10	4.82	4.78	5.14	4.87	4.83	5.11	4.85	4.81	6.54	5.11	5.04	6.13	7.57	5.79	6.3	**7.35**	**5.71**	**6.15**
IASI	*average*	*4.77*	*5.10*	*4.83*	*4.77*	*5.14*	*4.88*	*4.81*	*5.06*	*4.81*	*4.75*	*6.61*	*5.03*	*4.91*	*5.72*	*7.62*	*5.70*	*6.18*	*7.42*	*5.64*	*6.02*
AIRS	9	4.25	4.56	4.27	4.25	4.62	4.32	4.30	4.65	4.35	4.34	6.36	4.81	4.59	6.05	6.96	5.06	4.99	**7.07**	**5.17**	**5.04**
AIRS	16	4.22	4.51	4.23	4.22	4.57	4.28	4.28	4.60	4.31	4.31	6.22	4.71	4.53	5.84	6.73	4.92	4.94	**6.82**	**5.01**	**5**
AIRS	60	4.40	4.72	4.43	4.40	4.78	4.49	4.46	4.81	4.52	4.50	6.82	5.01	4.81	6.29	7.44	5.3	5.2	**7.53**	**5.41**	**5.27**
AIRS	82	4.15	4.45	4.16	4.15	4.50	4.21	4.20	4.53	4.24	4.23	5.85	4.59	4.38	5.63	6.32	4.76	4.83	**6.42**	**4.85**	**4.88**
AIRS	120	4.30	4.62	4.32	4.30	4.68	4.37	4.36	4.72	4.40	4.39	6.32	4.83	4.66	5.98	6.89	5.07	5.05	**6.98**	**5.18**	**5.12**
AIRS	126	4.42	4.74	4.45	4.42	4.80	4.51	4.48	4.82	4.53	4.51	6.70	5.02	4.79	6.29	7.31	5.3	5.2	**7.39**	**5.41**	**5.27**
AIRS	129	4.16	4.44	4.17	4.16	4.50	4.22	4.21	4.53	4.24	4.24	5.58	4.56	4.45	5.75	6.09	4.72	4.84	**6.17**	**4.81**	**4.9**
AIRS	151	4.43	4.74	4.46	4.43	4.80	4.52	4.49	4.82	4.53	4.51	6.45	4.98	4.81	6.48	6.99	5.19	5.18	**7.12**	**5.32**	**5.27**
AIRS	182	4.46	4.74	4.49	4.46	4.79	4.55	4.51	4.82	4.56	4.54	6.52	5.00	4.90	6.70	7.13	5.27	5.25	**7.21**	**5.38**	**5.32**
AIRS	193	4.44	4.76	4.47	4.44	4.82	4.53	4.50	4.86	4.56	4.54	6.65	5.03	4.84	6.29	7.23	5.29	5.22	**7.34**	**5.4**	**5.29**
AIRS	*average*	*4.32*	*4.63*	*4.34*	*4.32*	*4.69*	*4.40*	*4.38*	*4.72*	*4.43*	*4.41*	*6.35*	*4.85*	*4.68*	*6.13*	*6.91*	*5.09*	*5.07*	*7.01*	*5.19*	*5.14*
CRISM-FRT	frt00006d1c_07_sc164	4.76	4.76	5.69	8.42	4.80	5.68	8.41	4.80	5.69	8.53	4.88	5.34	8.21	9.06	6.18	6.44	6.49	**5.39**	**5.67**	**5.79**
CRISM-FRT	frt00010f86_07_sc167	5.43	5.43	7.00	9.50	5.48	7.00	9.51	5.47	7.09	9.73	5.55	6.06	9.46	10.02	6.99	7.39	7.43	**6.23**	**6.64**	**6.82**
CRISM-FRT	frt00013e49_07_sc166	5.62	5.62	7.36	9.54	5.66	7.36	9.54	5.63	7.44	9.77	6.29	6.12	9.44	10.08	7.49	7.45	7.53	**6.93**	**6.75**	**6.99**
CRISM-FRT	frt0000065e6_07_sc164	4.98	4.98	5.95	8.48	5.01	5.95	8.47	5.01	5.92	8.59	5.34	5.49	8.37	9.20	6.5	6.54	6.56	**5.84**	**5.8**	**5.89**
CRISM-FRT	frt0000869b_07_sc164	5.20	5.20	6.83	8.81	5.24	6.84	8.81	5.22	6.79	8.94	6.29	5.66	8.73	9.49	7.22	6.86	6.89	**6.76**	**6.14**	**6.29**
CRISM-FRT	frt0001077d_07_sc166	5.54	5.54	7.06	9.47	5.59	7.07	9.48	5.56	7.15	9.69	5.84	6.08	9.36	10.01	7.14	7.37	7.4	**6.5**	**6.66**	**6.85**
CRISM-FRT	frt0001241d_07_sc164	5.51	5.51	6.93	9.29	5.54	6.94	9.29	5.52	6.97	9.48	5.92	6.01	9.19	9.89	7.19	7.3	7.32	**6.54**	**6.56**	**6.72**
CRISM-FRT	frt00008849_07_sc165	5.04	5.04	6.33	8.78	5.08	6.33	8.78	5.06	6.30	8.91	5.52	5.53	9.19	9.45	6.74	6.75	6.8	**6.07**	**5.98**	**6.14**
CRISM-FRT	frt00009326_07_sc167	4.79	4.79	6.62	8.54	4.83	6.62	8.53	4.80	6.57	8.66	5.99	5.22	8.35	9.16	6.91	6.49	6.64	**6.48**	**5.76**	**6.09**
CRISM-FRT	*average (FRT)*	*5.21*	*5.21*	*6.64*	*8.98*	*5.25*	*6.64*	*8.98*	*5.23*	*6.66*	*9.15*	*5.74*	*5.73*	8.92	*9.60*	*6.93*	*6.95*	*7.01*	*6.30*	*6.22*	*6.40*
CRISM-HRL	hrl0000ba9c_07_sc183	5.09	5.09	6.32	8.98	5.11	6.32	8.99	5.10	6.36	9.19	5.53	5.64	8.91	9.58	6.81	6.89	6.97	**6.23**	**6.23**	**6.41**
CRISM-HRL	hrl00004f38_07_sc181	4.44	4.44	6.31	8.01	4.45	6.30	8.00	4.41	6.24	8.12	6.21	4.84	7.90	8.73	7	6.03	6.19	**6.65**	**5.36**	**5.69**
CRISM-HRL	hrl0000089fd_07_sc182	4.76	4.76	5.84	8.57	4.78	5.85	8.57	4.76	5.83	8.71	5.26	5.26	8.49	9.25	6.55	6.5	6.58	**5.92**	**5.79**	**5.95**
CRISM-HRL	hrl000064bf_07_sc182	4.63	4.63	6.26	8.30	4.64	6.25	8.29	4.62	6.20	8.43	5.98	5.08	8.22	9.03	6.91	6.3	6.42	**6.49**	**5.61**	**5.85**
CRISM-HRL	*average (HRL)*	*4.73*	*4.73*	*6.18*	*8.46*	*4.74*	*6.18*	*8.46*	*4.72*	*6.16*	*8.61*	*5.75*	*5.21*	8.38	*9.15*	*6.82*	*6.43*	*6.54*	*6.32*	*5.75*	*5.98*
CRISM-MSP	msp00003e34_01_sc214	2.73	2.73	3.39	6.55	2.63	3.33	6.57	2.69	3.37	6.98	3.22	3.25	6.68	7.21	4.63	4.63	5.16	**4.22**	**4.22**	**4.73**
CRISM-MSP	msp00003ea5_07_sc214	2.73	2.73	3.63	6.52	2.63	3.58	6.54	2.69	3.60	6.97	3.52	3.30	6.63	7.19	4.78	4.65	5.26	**4.46**	**4.27**	**4.92**
CRISM-MSP	msp00003fbc_03_sc214	2.77	2.77	3.95	6.45	2.67	3.92	6.48	2.74	3.90	6.87	3.95	3.35	6.60	7.19	5.07	4.67	5.32	**4.85**	**4.31**	**4.96**
CRISM-MSP	msp0000426b_01_sc214	2.70	2.70	3.17	6.52	2.60	3.09	6.55	2.66	3.14	6.93	2.96	3.21	6.68	7.02	4.51	4.6	5.02	**3.94**	**4.15**	**4.43**
CRISM-MSP	msp0000438d_03_sc214	2.72	2.72	3.71	6.46	2.62	3.67	6.48	2.69	3.66	6.89	3.67	3.28	6.58	7.11	4.88	4.63	5.3	**4.58**	**4.24**	**4.88**
CRISM-MSP	msp0001081c_07_sc214	3.70	3.70	4.45	7.37	3.59	4.38	7.39	3.78	4.59	8.15	4.11	4.28	7.61	8.02	5.49	5.63	6.03	**5.14**	**5.29**	**5.66**
CRISM-MSP	msp00004125_05_sc214	2.95	2.95	4.72	6.56	2.86	4.71	6.58	2.92	4.72	6.95	4.83	3.66	6.79	7.47	5.74	4.94	5.34	**5.72**	**4.68**	**5.04**
CRISM-MSP	*average (MSP)*	*2.90*	*2.90*	*3.86*	*6.63*	*2.80*	*3.81*	*6.66*	*2.88*	*3.86*	*7.11*	*3.75*	*3.47*	6.80	*7.32*	*5.01*	*4.82*	*5.35*	*4.70*	*4.45*	*4.95*
M3-Global	globalA	2.40	2.40	6.17	7.01	2.41	6.20	6.95	2.38	5.24	6.73	4.99	2.94	6.49	7.33	5.77	4.32	4.53	**5.44**	**3.59**	**3.98**
M3-Global	globalB	2.35	2.35	6.15	6.99	2.37	6.19	6.93	2.34	5.16	6.70	4.87	2.89	6.41	7.26	5.71	4.28	4.5	**5.36**	**3.54**	**3.89**
M3-Global	*average (global)*	*2.37*	*2.37*	*6.16*	*7.00*	*2.39*	*6.19*	*6.94*	*2.36*	*5.20*	*6.71*	*4.93*	*2.91*	6.45	*7.30*	*5.74*	*4.30*	*4.52*	*5.40*	*3.57*	*3.94*
M3-Target	targetA	3.53	3.53	5.27	7.01	3.54	5.22	6.99	3.53	4.50	6.89	3.99	3.81	6.88	7.77	5.25	4.85	4.95	**4.5**	**4.1**	**4.22**
M3-Target	targetB	3.33	3.33	5.18	6.70	3.35	5.14	6.67	3.34	4.45	6.61	4.08	3.62	6.70	7.65	5.14	4.65	4.72	**4.48**	**3.9**	**4.02**
M3-Target	targetC	2.88	2.88	5.06	6.49	2.91	4.98	6.45	2.90	4.07	6.32	3.37	3.24	6.22	7.12	4.7	4.33	4.44	**3.88**	**3.55**	**3.69**
M3-Target	*average (target)*	*3.25*	*3.25*	*5.17*	*6.73*	*3.27*	*5.11*	*6.70*	*3.26*	*4.34*	*6.61*	*3.82*	*3.56*	6.60	*7.51*	*5.03*	*4.61*	*4.70*	*4.29*	*3.85*	*3.98*
Hyperion	ErtaAle	4.33	4.36	4.66	5.22	4.40	4.70	5.26	4.49	4.78	5.46	5.04	4.65	5.47	6.14	5.55	5.03	5.03	**5.33**	**4.76**	**4.79**
Hyperion	LakeMonona	4.46	4.49	4.75	5.43	4.54	4.79	5.46	4.63	4.87	5.68	5.02	4.78	5.73	6.41	5.55	5.16	5.19	**5.34**	**4.88**	**4.97**
Hyperion	MtStHelens	4.38	4.40	4.77	5.25	4.44	4.81	5.29	4.55	4.95	5.51	5.25	4.71	5.52	6.14	5.73	5.12	5.13	**5.55**	**4.87**	**4.93**
Hyperion	Cuprite	4.33	4.35	4.63	5.09	4.40	4.67	5.13	4.48	4.74	5.33	5.05	4.62	5.35	5.96	5.56	4.96	4.97	**5.34**	**4.7**	**4.76**
Hyperion	*average (raw)*	*4.37*	*4.40*	*4.70*	*5.25*	*4.44*	*4.74*	*5.29*	*4.54*	*4.84*	*5.49*	*5.09*	*4.69*	*5.52*	*6.16*	*5.60*	*5.07*	*5.08*	*5.39*	*4.80*	*4.86*
Hyperion	Cuprite_flatfield	3.98	4.29	3.98	4.01	4.34	4.03	4.06	4.42	4.09	4.12	4.81	4.34	4.11	4.55	5	4.4	4.33	**5.08**	**4.4**	**4.27**
SFSI	mantar_raw	4.69	4.87	4.69	4.77	4.91	4.72	4.81	5.05	4.73	4.94	4.77	5.03	4.91	5.43	5.3	4.98	4.91	**4.79**	**4.72**	**4.67**
SFSI	mantar_rad_rmnoise	3.01	3.01	3.74	3.83	2.90	3.67	3.77	3.60	4.59	4.77	4.40	3.16	4.07	4.80	4.89	3.93	3.76	**4.56**	**3.51**	**3.38**
AVIRIS	sc0_raw	6.21	6.54	6.21	6.21	6.58	6.25	6.25	6.81	6.46	6.48	9.19	6.99	6.45	7.81	9.31	7.14	6.71	**9.54**	**7.21**	**6.72**
AVIRIS	sc3_raw	6.08	6.40	6.08	6.07	6.43	6.11	6.11	6.66	6.33	6.34	8.89	6.87	6.30	7.54	9.07	7.01	6.55	**9.27**	**7.08**	**6.55**
AVIRIS	sc10_raw	5.59	5.92	5.59	5.60	5.95	5.62	5.63	6.18	5.83	5.85	7.32	6.20	5.62	6.68	7.47	6.19	5.94	**7.57**	**6.19**	**5.9**
AVIRIS	sc11_raw	5.85	6.17	5.85	5.85	6.21	5.88	5.89	6.42	6.08	6.10	8.50	6.52	6.03	7.27	8.65	6.66	6.36	**8.83**	**6.68**	**6.35**
AVIRIS	sc18_raw	6.22	6.53	6.22	6.20	6.56	6.24	6.22	6.79	6.48	6.47	9.31	6.98	6.39	7.79	9.43	7.12	6.77	**9.67**	**7.2**	**6.8**
AVIRIS	*average (16-bit raw)*	*5.99*	*6.31*	*5.99*	*5.99*	*6.35*	*6.02*	*6.02*	*6.57*	*6.24*	*6.25*	*8.64*	*6.71*	*6.16*	*7.42*	*8.79*	*6.82*	*6.47*	*8.98*	*6.87*	*6.46*
AVIRIS	hawaii_sc01_raw	2.64	2.94	2.64	2.65	3.00	2.69	2.70	3.06	2.77	2.78	4.60	3.29	2.94	3.43	4.78	3.51	3.3	**4.7**	**3.34**	**3.08**
AVIRIS	maine_sc10_raw	2.74	3.04	2.74	2.75	3.08	2.79	2.79	3.17	2.87	2.88	4.53	3.39	3.07	3.64	4.67	3.62	3.36	**4.64**	**3.48**	**3.17**
AVIRIS	*average (12-bit raw)*	*2.69*	*2.99*	*2.69*	*2.70*	*3.04*	*2.74*	*2.74*	*3.11*	*2.82*	*2.83*	*4.56*	*3.34*	*3.01*	*3.53*	*4.73*	*3.57*	*3.33*	*4.67*	*3.41*	*3.13*
AVIRIS	sc0_cal	3.97	4.29	3.97	3.98	4.33	4.01	4.03	4.59	4.25	4.28	6.94	4.75	4.59	5.05	7.11	4.86	4.64	**7.22**	**4.81**	**4.54**
AVIRIS	sc3_cal	3.85	4.15	3.85	3.86	4.19	3.88	3.89	4.45	4.13	4.15	6.67	4.65	4.46	4.83	6.86	4.76	4.51	**6.95**	**4.7**	**4.4**
AVIRIS	sc10_cal	3.39	3.71	3.39	3.40	3.74	3.43	3.43	4.05	3.71	3.73	5.17	4.03	3.80	4.11	5.3	4.04	3.94	**5.28**	**3.92**	**3.77**
AVIRIS	sc11_cal	3.66	3.98	3.66	3.67	4.02	3.70	3.71	4.28	3.94	3.97	6.23	4.30	4.18	4.67	6.42	4.42	4.31	**6.47**	**4.32**	**4.18**
AVIRIS	sc18_cal	3.93	4.24	3.93	3.93	4.28	3.96	3.96	4.57	4.24	4.26	7.01	4.70	4.56	5.09	7.21	4.82	4.68	**7.32**	**4.77**	**4.59**
AVIRIS	*average (16-bit cal)*	*3.76*	*4.08*	*3.76*	*3.77*	*4.11*	*3.80*	*3.81*	*4.39*	*4.06*	*4.08*	*6.41*	*4.49*	4.32	*4.75*	*6.58*	*4.58*	*4.42*	*6.65*	*4.50*	*4.30*
CASI	t0180f07_raw	4.87	5.16	4.89	4.87	5.19	4.92	4.90	5.29	5.03	5.01	6.50	5.25	4.96	5.76	6.67	5.31	5.37	**6.71**	**5.24**	**5.29**
CASI	t0477f06_raw	5.20	5.41	5.28	5.20	5.43	5.31	5.23	5.51	5.39	5.29	7.08	5.46	5.23	6.15	7.4	5.62	5.75	**7.46**	**5.56**	**5.69**
CASI	*average (raw)*	*5.03*	*5.28*	*5.08*	*5.03*	*5.31*	*5.11*	*5.07*	*5.40*	*5.21*	*5.15*	*6.79*	*5.36*	*5.10*	*5.96*	*7.04*	*5.47*	*5.56*	*7.09*	*5.40*	*5.49*
CASI	t0180f07_rad	7.59	7.96	7.59	7.62	8.00	7.63	7.66	8.24	7.84	7.90	9.24	7.93	7.79	8.73	9.42	7.97	8.07	**9.57**	**7.98**	**8.11**
CASI	t0477f06_rad_rss	8.31	8.62	8.31	8.30	8.65	8.34	8.33	8.79	8.46	8.49	10.37	8.59	8.56	9.80	10.61	8.74	8.82	**10.82**	**8.81**	**8.92**

Table E-4: Compressed Data Rates (in Bits/Sample) for Segmented Compression of Multispectral Images

		CCSDS-123.0-B										JPEG-LS		ESA + sample-adaptive coding	LUT + sample-adaptive coding	CCSDS-122.0-B			JPEG2000		
		defaults	sample-adaptive			block-adaptive BIL			block-adaptive BIP												
Instrument	scene		reduced+ column	full+ neighbor	reduced+ neighbor	reduced+ column	full+ neighbor	reduced+ neighbor	reduced+ column	full+ neighbor	reduced+ neighbor	direct	differential			direct	+ IWT	+ POT	direct	+ IWT	+ POT
MODIS-night	A2001123.0000_night	4.35	4.35	4.88	6.94	4.38	4.90	6.99	4.94	5.22	7.26	4.96	5.36	7.69	6.94	5.89	6.04	4.72	**5.14**	**5.27**	**4.55**
MODIS-night	A2001123.1630_night	5.06	5.06	5.69	7.20	5.07	5.71	7.24	5.93	6.27	7.50	5.98	6.44	7.88	7.67	6.61	6.85	5.63	**6.12**	**6.4**	**5.56**
MODIS-night	A2001222.0835_night	4.38	4.38	4.90	7.03	4.39	4.91	7.07	4.96	5.25	7.39	5.00	5.44	7.63	6.96	5.9	6.13	4.83	**5.13**	**5.36**	**4.65**
MODIS-night	A2001222.0840_night	5.10	5.10	5.74	7.32	5.13	5.77	7.36	6.12	6.41	7.68	6.10	6.59	7.96	7.73	6.76	7.05	5.71	**6.26**	**6.61**	**5.63**
MODIS-night	A2001222.1200_night	4.32	4.32	4.80	7.01	4.34	4.81	7.05	5.03	5.24	7.40	4.90	5.42	7.62	6.92	5.84	6.13	4.85	**5.05**	**5.36**	**4.67**
MODIS-night	***average (night)***	***4.64***	***4.64***	***5.20***	***7.10***	***4.66***	***5.22***	***7.14***	***5.40***	***5.68***	***7.45***	***5.39***	***5.85***	***7.76***	***7.24***	***6.20***	***6.44***	***5.15***	***5.54***	***5.80***	***5.01***
MODIS-day	A2001123.0000_day	4.13	4.13	4.13	4.20	3.44	3.43	3.50	6.45	6.17	6.50	2.83	3.07	4.08	4.69	3.51	5.01	4.67	**3.35**	**4.94**	**4.3**
MODIS-day	A2001123.1630_day	5.96	5.96	5.98	6.04	5.65	5.67	5.74	7.27	7.29	7.46	5.08	5.59	6.02	7.04	5.91	7.13	7.15	**5.8**	**7.08**	**7.04**
MODIS-day	A2001222.0835_day	6.26	6.26	6.41	6.48	5.87	6.06	6.15	6.90	7.03	7.21	5.39	5.67	6.48	7.40	6.44	7.07	6.99	**6.22**	**6.95**	**6.82**
MODIS-day	A2001222.0840_day	6.15	6.15	6.13	6.16	5.89	5.87	5.90	7.56	7.48	7.59	5.13	5.67	6.26	6.77	6.18	7.42	7.17	**6.01**	**7.39**	**7**
MODIS-day	A2001222.1200_day	5.66	5.66	5.82	5.85	5.22	5.42	5.46	6.63	6.82	6.98	5.04	5.19	5.91	6.82	5.9	6.61	6.64	**5.76**	**6.48**	**6.43**
MODIS-day	***average (day)***	***5.63***	***5.63***	***5.69***	***5.75***	***5.21***	***5.29***	***5.35***	***6.96***	***6.96***	***7.15***	***4.69***	***5.04***	**5.75**	***6.54***	***5.59***	***6.65***	***6.52***	***5.43***	***6.57***	***6.32***
MODIS-500m	A2001123.0000_500m	6.83	7.40	7.17	7.77	7.44	7.20	7.81	8.55	7.42	8.37	7.57	6.39	8.38	8.97	8.12	7.88	7.19	**7.73**	**6.73**	**6.57**
MODIS-500m	A2001123.1630_500m	7.51	7.96	7.63	8.07	7.95	7.62	8.08	8.91	7.98	8.68	7.95	7.08	8.48	9.11	8.42	8.37	7.85	**8.13**	**7.61**	**7.43**
MODIS-500m	A2001222.0835_500m	6.89	7.52	6.95	7.53	7.53	6.95	7.55	8.81	7.45	8.51	7.18	6.27	8.11	8.51	7.88	7.97	7.33	**7.28**	**6.78**	**6.53**
MODIS-500m	A2001222.0840_500m	7.67	8.19	7.66	8.11	8.21	7.67	8.14	9.14	8.12	8.94	8.07	7.24	8.56	9.14	8.54	8.57	8.06	**8.29**	**7.93**	**7.65**
MODIS-500m	A2001222.1200_500m	7.03	7.60	7.14	7.73	7.60	7.13	7.74	8.90	7.56	8.68	7.40	6.30	8.28	8.67	8.11	8.02	7.39	**7.55**	**6.76**	**6.57**
MODIS-500m	***average (500m)***	***7.19***	***7.73***	***7.31***	***7.84***	***7.75***	***7.31***	***7.86***	***8.86***	***7.71***	***8.64***	***7.63***	***6.66***	**8.36**	***8.88***	***8.21***	***8.16***	***7.56***	***7.80***	***7.16***	***6.95***
MODIS-250m	A2001123.0000_250m	5.83	6.12	6.03	6.31	6.12	6.04	6.33	6.52	6.18	6.56	6.88	6.29	6.79	6.85	6.95	6.59	5.64	**6.97**	**6.54**	**5.44**
MODIS-250m	A2001123.1630_250m	7.02	7.33	6.99	7.25	7.26	6.93	7.20	7.42	7.05	7.34	7.20	7.00	7.44	8.35	7.36	7.37	7.2	**7.37**	**7.36**	**7.19**
MODIS-250m	A2001222.0835_250m	6.06	6.29	6.24	6.69	6.24	6.21	6.69	6.51	6.36	6.83	6.66	6.36	6.86	7.51	6.84	6.86	6.25	**6.73**	**6.74**	**6.18**
MODIS-250m	A2001222.0840_250m	7.29	7.62	7.28	7.59	7.59	7.27	7.58	7.81	7.45	7.75	7.45	7.23	7.90	8.54	7.67	7.84	7.52	**7.71**	**7.89**	**7.58**
MODIS-250m	A2001222.1200_250m	5.94	6.16	6.25	6.66	6.10	6.21	6.66	6.36	6.27	6.75	6.61	6.25	6.78	7.41	6.87	6.71	5.9	**6.77**	**6.56**	**5.78**
MODIS-250m	***average (250m)***	***6.43***	***6.70***	***6.56***	***6.90***	***6.66***	***6.53***	***6.89***	***6.92***	***6.66***	***7.05***	***6.96***	***6.63***	**7.15**	***7.73***	***7.14***	***7.07***	***6.50***	***7.11***	***7.02***	***6.43***
MSG	3	3.34	3.78	3.34	3.55	3.74	3.30	3.50	4.06	3.58	3.81	3.44	3.80	4.06	5.01	3.62	4.04	3.86	**3.46**	**3.89**	**3.69**
MSG	15	3.54	4.02	3.54	3.74	3.99	3.51	3.70	4.29	3.77	4.00	3.70	3.96	4.26	5.25	3.89	4.17	4.02	**3.72**	**3.99**	**3.83**
MSG	31	3.31	3.75	3.31	3.52	3.71	3.27	3.47	3.99	3.51	3.73	3.37	3.74	4.01	4.93	3.58	3.97	3.81	**3.41**	**3.81**	**3.63**
MSG	***average***	***3.40***	***3.85***	***3.40***	***3.60***	***3.81***	***3.36***	***3.56***	***4.11***	***3.62***	***3.85***	***3.50***	***3.83***	***4.11***	***5.07***	***3.70***	***4.06***	***3.90***	***3.53***	***3.90***	***3.72***
LANDSAT	agriculture	3.60	3.97	3.60	3.71	3.98	3.62	3.73	3.99	3.65	3.77	3.99	3.81	4.51	4.82	4.38	4.07	3.99	**4.29**	**3.94**	**3.83**
LANDSAT	coast	2.74	3.06	2.74	2.76	3.06	2.74	2.76	3.11	2.76	2.79	2.94	2.97	2.89	3.50	3.25	3.16	3.17	**3.05**	**2.9**	**2.92**
LANDSAT	mountain	3.78	4.15	3.78	3.84	4.17	3.80	3.87	4.39	4.00	4.11	4.17	4.08	4.32	4.65	4.45	4.37	4.25	**4.35**	**4.23**	**4.12**
LANDSAT	***average***	***3.37***	***3.73***	***3.37***	***3.44***	***3.74***	***3.39***	***3.45***	***3.83***	***3.47***	***3.56***	***3.70***	***3.62***	***3.91***	***4.32***	***4.03***	***3.87***	***3.80***	***3.90***	***3.69***	***3.62***
PLEIADES	montp_trans	7.43	7.77	7.43	7.57	7.83	7.48	7.63	8.16	7.75	7.95	8.00	7.42	8.05	8.76	8.2	7.76	8.02	**8.4**	**7.93**	**8.2**
PLEIADES	perp_trans	7.22	7.52	7.22	7.37	7.53	7.24	7.39	7.70	7.37	7.57	7.70	7.24	7.85	8.71	7.89	7.4	7.64	**8.09**	**7.56**	**7.81**
PLEIADES	***average***	***7.32***	***7.65***	***7.32***	***7.47***	***7.68***	***7.36***	***7.51***	***7.93***	***7.56***	***7.76***	***7.85***	***7.33***	***7.95***	***8.74***	***8.05***	***7.58***	***7.83***	***8.25***	***7.75***	***8.01***
PLEIADES	montp_misreg0	6.72	7.03	6.72	6.85	7.08	6.76	6.89	7.48	7.12	7.30	7.26	6.68	7.31	8.02	7.46	7.1	7.28	**7.6**	**7.21**	**7.39**
PLEIADES	montp_misreg20	6.98	7.27	6.98	7.12	7.31	7.02	7.16	7.64	7.30	7.48	7.25	6.90	7.53	8.19	7.46	7.26	7.48	**7.6**	**7.38**	**7.6**
PLEIADES	montp_misreg35	7.14	7.44	7.14	7.29	7.46	7.17	7.32	7.74	7.40	7.58	7.26	7.07	7.69	8.33	7.46	7.37	7.59	**7.6**	**7.49**	**7.72**
PLEIADES	montp_misreg50	7.26	7.57	7.26	7.43	7.59	7.29	7.45	7.82	7.47	7.66	7.26	7.21	7.83	8.46	7.46	7.44	7.66	**7.6**	**7.56**	**7.79**
VEGETATION	1_1b	5.24	5.28	5.24	5.72	5.20	5.17	5.62	5.33	5.30	6.00	5.30	5.05	5.82	6.93	5.56	5.52	5.5	**5.51**	**5.42**	**5.38**
VEGETATION	2_1b	5.28	5.32	5.28	5.76	5.24	5.21	5.66	5.37	5.35	6.03	5.33	5.07	5.90	7.00	5.61	5.56	5.56	**5.55**	**5.47**	**5.44**
VEGETATION	***average (level 1b)***	***5.26***	***5.30***	***5.26***	***5.74***	***5.22***	***5.19***	***5.64***	***5.35***	***5.32***	***6.02***	***5.31***	***5.06***	***5.86***	***6.97***	***5.59***	***5.54***	***5.53***	***5.53***	***5.45***	***5.41***
VEGETATION	1_1c	5.02	5.29	5.02	5.14	5.21	4.95	5.06	5.34	5.03	5.18	5.25	5.02	5.51	6.73	5.44	5.32	5.25	**5.44**	**5.32**	**5.24**
VEGETATION	2_1c	5.06	5.33	5.06	5.18	5.25	4.99	5.11	5.38	5.08	5.23	5.29	5.04	5.58	6.81	5.48	5.37	5.33	**5.48**	**5.36**	**5.32**
VEGETATION	***average (level 1c)***	***5.04***	***5.31***	***5.04***	***5.16***	***5.23***	***4.97***	***5.08***	***5.36***	***5.06***	***5.21***	***5.27***	***5.03***	**5.55**	***6.77***	***5.46***	***5.35***	***5.29***	***5.46***	***5.34***	***5.28***
SPOT5	1	5.14	5.48	5.14	5.25	5.48	5.15	5.26	5.50	5.17	5.30	5.45	5.39	5.63	6.04	5.64	5.7	5.34	**5.63**	**5.7**	**5.3**
SPOT5	2	4.28	4.64	4.28	4.37	4.63	4.26	4.35	4.65	4.28	4.39	4.36	4.40	4.89	5.44	4.63	4.79	4.4	**4.61**	**4.77**	**4.33**
SPOT5	3	4.17	4.60	4.17	4.34	4.60	4.17	4.34	4.60	4.17	4.35	4.33	4.36	4.99	5.52	4.61	4.73	4.36	**4.59**	**4.7**	**4.28**
SPOT5	***average***	***4.53***	***4.91***	***4.53***	***4.66***	***4.90***	***4.53***	***4.65***	***4.92***	***4.54***	***4.68***	***4.71***	***4.72***	***5.17***	***5.66***	***4.96***	***5.07***	***4.70***	***4.94***	***5.06***	***4.64***

ANNEX F

ABBREVIATIONS AND ACRONYMS

AIRS	Atmospheric Infrared Sounder
AVIRIS	Airborne Visible/Infrared Imaging Spectrometer
BI	band interleaved
BIL	band-interleaved lines
BIP	band-interleaved pixels
BSQ	band sequential
CASI	Compact Airborne Spectrographic Imager
CFDP	CCSDS File Delivery Protocol
CRISM	Compact Reconnaissance Imaging Spectrometer for Mars
DN	digital number
DPCM	differential pulse code modulation
DWTs	discrete wavelet transforms
FL	fast lossless
FPGA	field programmable gate array
FRT	full-resolution target
GPO2	Golomb-power-of-2
HR	high resolution
HRG	high resolution geometric
HRL	half-resolution long
IASI	Infrared Atmospheric Sounding Interferometer
ISO	International Organization for Standardization
ITU	International Telecommunication Union
IWT	integer wavelet transform
LMS	least mean square
LUT	lookup table
M3	Moon Mineralogy Mapper

MAD	maximum absolute difference
MAE	mean absolute error
MCT	multi-component transform
MODIS	Moderate Resolution Imaging Spectroradiometer
MSG	Meteosat Second Generation
MSP	multispectral survey
POT	pairwise orthogonal transform
PSNR	peak signal-to-noise ratio
PVN	packet version number
RMS	root mean square
RMSE	root mean square error
ROI	region-of-interest
RRMSE	relative RMSE
SFSI	SWIR Full Spectrum Imager
SNR	signal-to-noise ratio
SWIR	short wave infrared