The Consultative Committee for Space Data Systems

Report Concerning Space Data System Standards

LOW-COMPLEXITY LOSSLESS AND NEAR-LOSSLESS MULTISPECTRAL AND HYPERSPECTRAL IMAGE COMPRESSION

INFORMATIONAL REPORT

CCSDS 120.2-G-2

GREEN BOOK

December 2022

The Consultative Committee for Space Data Systems

Report Concerning Space Data System Standards

LOW-COMPLEXITY LOSSLESS AND NEAR-LOSSLESS MULTISPECTRAL AND HYPERSPECTRAL IMAGE COMPRESSION

INFORMATIONAL REPORT

CCSDS 120.2-G-2

GREEN BOOK

December 2022

AUTHORITY

Issue:	Informational Report, Issue 2
Date:	December 2022
Location:	Washington, DC, USA

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

This document is published and maintained by:

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

FOREWORD

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

http://www.ccsds.org/

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

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

Member Agencies

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

Observer Agencies

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

DOCUMENT CONTROL

Document	Title	Date	Status
CCSDS 120.2-G-1	Lossless Multispectral and Hyperspectral Image Compression, Informational Report, Issue 1	December 2015	Original issue, superseded
CCSDS 120.2-G-2	Low-Complexity Lossless and Near-Lossless Multispectral and Hyperspectral Image Compression, Informational Report, Issue 2	December 2022	Current issue

CONTENTS

CONTENTS (continued)

CONTENTS (continued)

Figure Page

CONTENTS (continued)

Figure Page

CONTENTS (continued)

Figure Page

CONTENTS (continued)

Figure Page

Table

CONTENTS (continued)

1 INTRODUCTION

1.1 PURPOSE

This report presents a summary of the key operational concepts and rationale that underlie the requirements for the CCSDS Recommended Standard, *Low-Complexity Lossless and Near-Lossless Multispectral and Hyperspectral Image Compression* (reference [1]). Supporting performance information, along with illustrations, is also included. This report provides a broad tutorial overview of the 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 *Low-Complexity Lossless and Near-Lossless Multispectral and 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.
- 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 provides detailed compression results.

– Annex E provides a list of abbreviations and acronyms used in the text of this document.

– Annex F gives a proof of a bound on a coding parameter as a function of bit depth.

1.4 CONVENTION

The following convention applies throughout this document:

– The capitalized phrase 'Recommended Standard' by itself refers to *Low-Complexity Lossless and Near-Lossless Multispectral and 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 publications are referenced in this document. At the time of publication, the editions indicated were valid. All publications are subject to revision, and users of this document are encouraged to investigate the possibility of applying the most recent editions of the publications indicated below. The CCSDS Secretariat maintains a register of currently valid CCSDS publications.

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

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

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

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

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

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

[7] *TM Synchronization and Channel Coding—Summary of Concept and Rationale*. Issue 3. Report Concerning Space Data System Standards (Green Book), CCSDS 130.1-G-3. Washington, D.C.: CCSDS, June 2020.

[8] *Lossless Multispectral & Hyperspectral Image Compression*. Issue 1-S. Recommendation for Space Data System Standards (Historical), CCSDS 123.0-B-1-S. Washington, D.C.: CCSDS, (May 2012) February 2019.

[9] 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.

[10] M. Klimesh. "Low-Complexity Adaptive Lossless Compression Of Hyperspectral Imagery." *SPIE Proceedings: Satellite Data Compression, Communications, and Archiving II* 6300 (September 2006): 63000N-1–63000N-9.

[11] D. Keymeulen, et al. "High Performance Space Data Acquisition and Compression with Embedded System-on-Chip Instrument Avionics for Space-Based Next Generation Imaging Spectrometers (NGIS)." In *Proceedings of 27th Annual Single Event Effects (SEE) Symposium and Military and Aerospace Programmable Logic Devices (MAPLD) Workshop (May 21–24, 2018, La Jolla, California)*. Pasadena, California: JPL, 2018.

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

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

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

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

[16] 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.

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

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

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

[20] P.G. Howard. "Interleaving Entropy Codes." In *Proceedings of Compression and Complexity of Sequences 1997 (13–13 June 1997, Salerno, Italy)*, 45–55. Piscataway, New Jersey: IEEE Conference Publications, 1997.

[21] A. Kiely and M. Klimesh. "A New Entropy Coding Technique for Data Compression." *IPN Progress Report* 42-146 (August 15, 2001).

[22] *Lossless Data Compression*. Issue 4. Report Concerning Space Data System Standards (Green Book), CCSDS 120.0-G-4. Washington, D.C.: CCSDS, November 2021.

[23] 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).

[24] I. Blanes, et al. "Performance Impact of Parameter Tuning on the Emerging CCSDS-123.0-B-2 Low-Complexity Lossless and Near-Lossless Multispectral and Hyperspectral Image Compression Standard." In *Proceedings of the 6th International Workshop on On-Board Payload Data Compression (20–21 September 2018, Matera, Italy)*. Noordwijk, The Netherlands: ESA Conference Bureau, 2018.

[25] I. Blanes, et al. "Performance Impact of Parameter Tuning on the CCSDS-123.0-B-2 Low-Complexity Lossless and Near-Lossless Multispectral and Hyperspectral Image Compression Standard." *Remote Sensing* 11, no. 11 (2019).

[26] 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.

[27] G. Blanchet, et al. "PLEIADES-HR Innovative Techniques for Radiometric Image Quality Commissioning." *International Archives of the Photogrammetry, Remote Sensing, and Spatial Information Science* 39, no. B1 (2012): 513–518.

[28] R. Camarero, et al. "Innovative Techniques in Predictive Lossy Compression for Future CNES Missions." 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.

[29] J.-L. Starck, F.D. Murtagh, and A. Bijaoui. *Image Processing and Data Analysis: The Multiscale Approach*. Cambridge: Cambridge University Press, 2009.

[30] D. Valsesia and E. Magli. "Fast and Lightweight Rate Control for Onboard Predictive Coding of Hyperspectral Images." *IEEE Geoscience and Remote Sensing Letters* 14, no. 3 (2017): 394–398.

[31] J. Bartrina-Rapesta, et al. "A Novel Rate-Control for Predictive Image Coding With Constant Quality." *IEEE Access* 7 (2019): 103918–103930.

[32] *Encapsulation Packet Protocol*. Issue 3. Recommendation for Space Data System Standards (Blue Book), CCSDS 133.1-B-3. Washington, D.C.: CCSDS, May 2020.

[33] *TM Synchronization and Channel Coding*. Issue 4. Recommendation for Space Data System Standards (Blue Book), CCSDS 131.0-B-4. Washington, D.C.: CCSDS, April 2022.

[34] *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.

[35] *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.

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

[37] J-M. Gaucel, et al. "On-Board Compression of Hyperspectral Satellite Data Using Band–Reordering." *SPIE Proceedings: Satellite Data Compression, Communications, and Processing VII* 8157 (September 16, 2011).

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

[39] 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.

[40] 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.

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

[42] 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.

[43] A. Rodríguez, et al. "Scalable Hardware-Based On-Board Processing for Run-Time Adaptive Lossless Hyperspectral Compression." *IEEE Access* 7 (2019): 10644–10652.

[44] L. Santos, A. Gómez, and R. Sarmiento. "Implementation of CCSDS Standards for Lossless Multispectral and Hyperspectral Satellite Image Compression." *IEEE Transactions on Aerospace and Electronic Systems* 56, no. 2 (2020): 1120–1138.

[45] A. Tsigkanos, et al. "A 3.3 Gbps CCSDS 123.0-B-1 Multispectral & Hyperspectral Image Compression Hardware Accelerator on a Space-Grade SRAM FPGA." *IEEE Transactions on Emerging Topics in Computing* 9, no. 1 : 90–103.

[46] D. Keymeulen, et al. "FPGA Implementation of Space-Based Lossless and Lossy Multispectral and Hyperspectral Image Compression." In *Proceedings of the 5th International Workshop on On-Board Payload Data Compression (28–29 September 2016, Frascati, Italy)*, 28–29. Noordwijk, The Netherlands: ESA Conference Bureau, 2016.

[47] G. Yu, T. Vladimirova, and M.N. Sweeting. "FPGA-Based On-Board Multi/Hyperspectral Image Compression System." In *Proceedings of the 2009 IEEE International Geoscience and Remote Sensing Symposium (12–17 July 2009, Cape Town, South Africa)*. 122.1-B-1. Piscataway, New Jersey: IEEE Conference Publications, 2009.

[48] *Spectral Preprocessing Transform for Multispectral and Hyperspectral Image Compression*. Issue 1. Recommendation for Space Data System Standards (Blue Book), CCSDS 122.1-B-1. Washington, D.C.: CCSDS, September 2017.

[49] *Image Data Compression*. Issue 2-S. Recommendation for Space Data System Standards (Blue Book), CCSDS 122.0-B-2. Washington, D.C.: CCSDS, September 2017.

[50] *Image Data Compression*. Issue 3. Report Concerning Space Data System Standards (Green Book), CCSDS 120.1-G-3. Washington, D.C.: CCSDS, November 2021.

[51] *Spectral Pre-Processing Transform for Multispectral & Hyperspectral Image Compression*. Issue 1. Report Concerning Space Data System Standards (Green Book), CCSDS 120.3-G-1. Washington, D.C.: CCSDS, March 2019.

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

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

[54] 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.

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

[56] 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.

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

2 OVERVIEW

2.1 INTRODUCTION

The CCSDS *Low-Complexity Lossless and Near-Lossless Multispectral and Hyperspectral Image Compression* Recommended Standard (reference [1]) defines for standardization a particular lossless and near-lossless data compressor that is applicable to three-dimensional images produced by multispectral and hyperspectral imagers and sounders. Here, 'near-lossless' refers to the ability to perform compression in a way that the maximum error in the reconstructed image can be limited to a user-specified bound.

The compressor is intended to be suitable for use on board spacecraft; in particular, the algorithm complexity and memory usage are designed to be sufficiently low to make high-speed and low-power hardware implementation feasible. The compressor provides other key features for onboard use, such as limiting the impact of a communication packet loss and providing error containment and robustness of its compression efficiency against detector or instrument defects.

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 32 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]), z,x,y (Band-Interleaved Pixels [BIP]), and x,z,y (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 that 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,

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

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

and so on, 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 either exactly or approximately. That is, the Recommended Standard can be used to provide lossless or near-lossless compression. The perfect fidelity required by lossless compression results in a lower compression ratio (i.e., higher volume of compressed data) for a given source image.

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 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 [2]);
- CCSDS File Delivery Protocol (CFDP) (reference [3]);
- packet service or bitstream service as provided by the AOS Space Data Link Protocol (reference [4]) or TM Space Data Link Protocol (reference [5]);
- Unified Space Data Link Protocol (ULSP) (reference [57]).

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 [6]).

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 reconstruction of a compressed image; the effects of a single bit error or loss of compressed image data can propagate to corrupt reconstructed data affecting the entire image, though in some cases partial recovery may be possible (see 5.3.1). Therefore measures should be taken to maximize the reliability of the data transmission link. Reference [7] gives 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 in order to limit the impact of data loss or corruption on the communications channel, limit the maximum possible size of a compressed image, and/or improve throughput of an implementation by compressing images in parallel. 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

Issue 1 of the Recommended Standard (reference [8]) is a formalization of the Fast Lossless (FL) compressor presented in references [9] and [10], which 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 [9] and [10]).

Because of the significant data volume reduction often needed to meet spacecraft downlink limitations, lossy compression is becoming increasingly used in space applications. With this motivation, the MHDC working group developed issue 2 of CCSDS-123.0-B, extending the standard's capabilities to provide low-complexity near-lossless compression while still supporting lossless compression. In this context, 'near-lossless' refers to the ability to perform compression in a way that limits the maximum error in the reconstructed image to a user-specified bound.

The present Recommended Standard, issue 2 (reference [1]), is based largely on the FL Extended (FLEX) compressor, which extends FL to provide adjustable near-lossless compression in addition to lossless compression and adds a new entropy coding option for better compression of low-entropy data (reference [11]). FLEX inherits many of the desirable features of the FL compressor, such as low computational complexity, single-pass compression, and automatic adaptation to the source image data.

Issue 2 incorporates an improvement of FLEX's hybrid entropy coder, and adds optional features such as relative error limits, periodic error limit updating, and narrow local sums (see 3.2).

The compressor estimates sample values by linear prediction, which is a natural strategy for lossless and near-lossless compression of multispectral and hyperspectral images. The differences between the estimates and the actual sample values are quantized, and the quantizer output is 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.

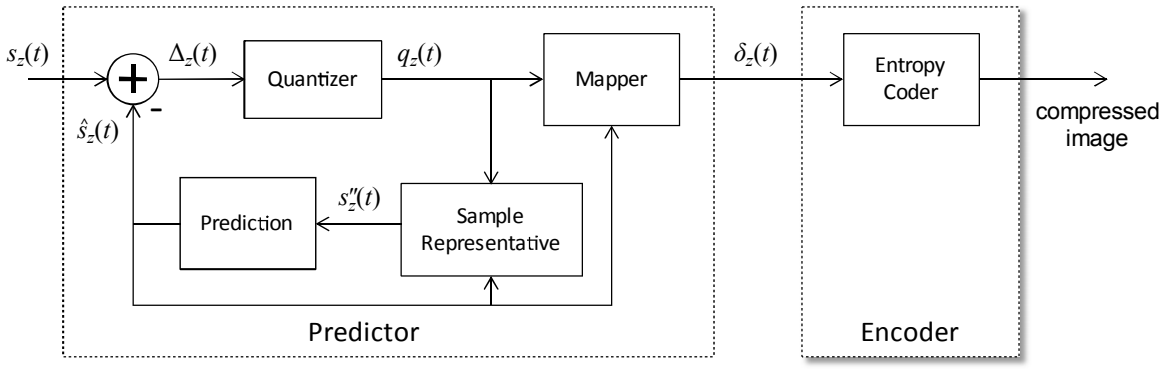


Figure 3-1: Compressor Schematic

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. Prediction can be performed causally in a single pass through the image. The prediction residual, that is, the difference between the predicted and actual sample values, is quantized, and the quantizer output is mapped to an unsigned integer that can be represented using the same number of bits as the input data sample. This mapping is invertible, so the decompressor can exactly reconstruct the quantizer index. These mapped quantizer indices 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 that losslessly encodes the mapped quantizer indices. Three different entropy coding options may be used for a given image. In each case, entropy coder parameters are adaptively adjusted during encoding to adapt to changes in the statistics of the mapped quantizer indices. Subsection 3.3 describes entropy coding 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 in 3.2.2. This description uses real-valued quantities and arithmetic. Subsection 3.2.3 describes how this algorithm can be converted to a version that uses only integer arithmetic but is capable of producing essentially equivalent predictions.

Subsection 3.2.4 describes modifications to the lossless predictor that provide near-lossless compression.

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 quantizer index $\delta_{z,y,x}$, defined in equation (55) 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 $z < P$.

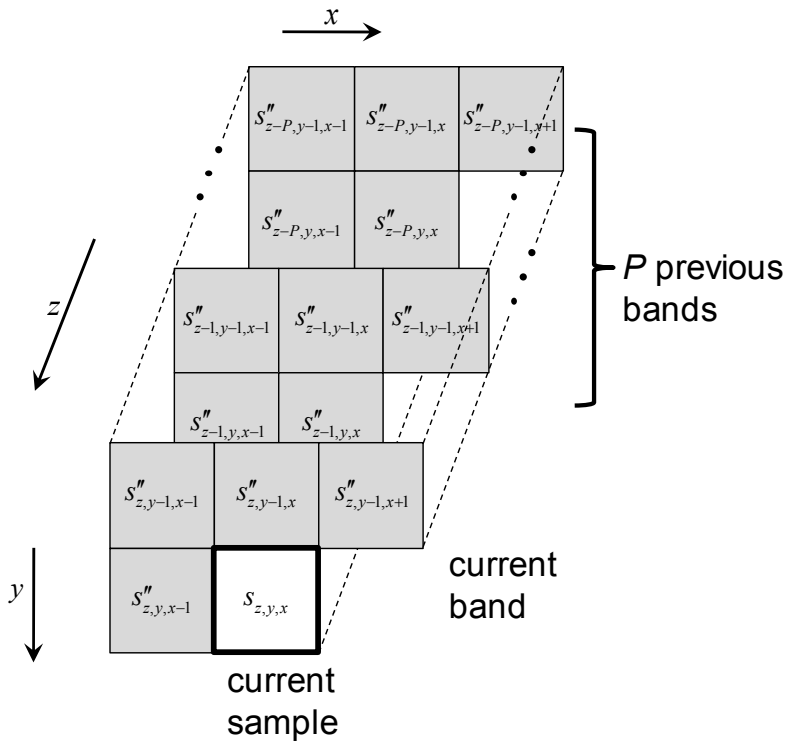


Figure 3-2: Typical Prediction Neighborhood

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).

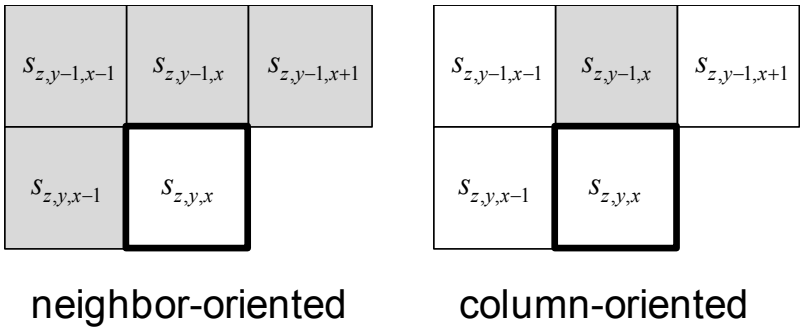


Figure 3-3: Calculating the Local Mean

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 to enable the same prediction framework to 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', that is, $s_{z,y-1,x+1} - \mu_{z,y,x}$, 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.

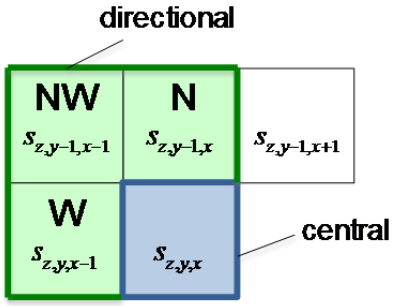


Figure 3-4: Minuends for Computing Directional and Central Local Differences in a Spectral Band

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 of $\mathbf{V}_z^{\mathrm{T}}(t)\Psi_{z,y,x}$.

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

Following the calculation of each predicted sample value, the weights used in prediction are adaptively updated using the sign algorithm. The sign algorithm (reference [12]) is a relative of the Least Mean Square (LMS) algorithm (references [13] and [14]), 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 will decrease 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 an integer version:

- 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 either an integer *wide local sum* $\sigma_{z,y,x} = 4\mu_{z,y,x}$, or a *narrow local sum*, which is similar but defined in a way that does not depend on $s_{z,y,x-1}$ (reference [1]).

- 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; 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. An additional user-specified *weight exponent offset* term is added to the exponent of each prediction weight update equation; these offsets are intended to better accommodate band-to-band signal energy variations in in the input image.

– 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 quantizer indices (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 4.10.3 of reference [1]) includes a factor of $\text{sgn}^+[e_z(t)]$ rather than $\text{sgn}[e_z(t)]$.

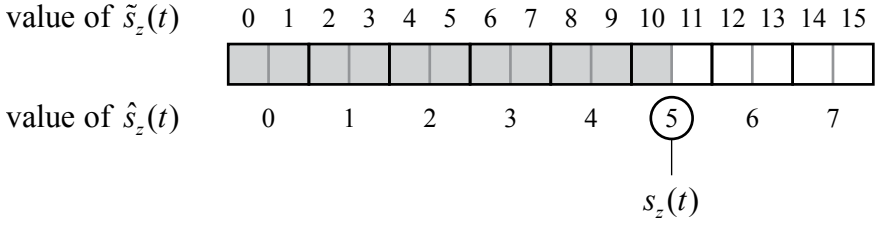


Figure 3-5: Relationship between $\tilde{s}_z(t)$ and $\hat{s}_z(t)$ for an Image with 3-Bit Unsigned Samples

3.2.4 EXTENDING THE FL PREDICTOR TO PROVIDE NEAR-LOSSLESS COMPRESSION

To provide near-lossless compression, the predictor is modified in two major respects. First, each prediction residual is quantized using a uniform quantizer with step size determined by user-specified fidelity parameters as described below in 3.2.4.1. Second, the predictor cannot utilize the original sample values $s_z(t)$, because these values will not be available to the decompressor at the time of reconstruction when compression is not lossless. Instead, prediction calculations are performed using a *sample representative* $s_z''(t)$ in place of each original sample value $s_z(t)$, as described in 3.2.4.2.

3.2.4.1 Quantization

The incorporation of an in-loop scalar quantizer in the predictor provides near-lossless compression. Quantizer fidelity settings can vary from band to band and can be updated periodically within the image.

User-specified *absolute* and/or *relative error limit* values control the *maximum error* value $m_z(t)$ for each sample. The prediction residual,

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

is quantized by a uniform quantizer centered at zero and having step size $2m_z(t) + 1$, which guarantees that the sample can be reconstructed with at most $m_z(t)$ units of error. The output of the quantizer is the signed integer *quantizer index* $q_z(t)$, defined as

$$q_z(t) = \begin{cases} \Delta_z(0), & t = 0 \\ \text{sgn}\big(\Delta_z(t)\big)\left\lfloor \dfrac{|\Delta_z(t)| + m_z(t)}{2m_z(t) + 1} \right\rfloor, & t > 0. \end{cases}$$

A quantizer index of zero indicates that the prediction was within $m_z(t)$ of the actual sample value. Positive (negative) quantizer indices correspond to cases in which the true sample value overshot (undershot) the prediction by more than $m_z(t)$.

Users can choose to use lossless compression, which sets $m_z(t) = 0$ for all z and t. Alternatively, users can control the maximum error value by specifying an *absolute error limit* a_z for each z, a *relative error limit* r_z for each z, or both.

When only absolute error limits are used, the maximum error is computed as

$$m_z(t) = a_z$$

for all z and t; when only relative error limits are used,

$$m_z(t) = \left\lfloor \frac{r_z |\hat{s}_z(t)|}{2^D} \right\rfloor$$

for all z and t,so that samples predicted to have smaller magnitude can be reconstructed with higher fidelity; and when both absolute and relative error limits are used,

$$m_z(t) = \min\left(a_z, \left\lfloor \frac{r_z |\hat{s}_z(t)|}{2^D} \right\rfloor \right)$$

for all z and t.

By varying the values of a_z and/or r_z from band to band, bands with higher science value can be preserved with higher fidelity (or losslessly) while reducing the data volume used to encode bands with lower science value.

Error limit values may be fixed for an entire image, or the user may choose to use *periodic error limit updating*, in which case, new error limit values are periodically encoded every 2^u frames, where u is a user-specified parameter, in the compressed bitstream. This capability could be used, for example, to provide higher fidelity data for regions of an image that are expected to contain features of interest, or to adaptively adjust fidelity parameters to meet a downlink rate constraint. The Recommended Standard does not specify a particular method for selecting error limit values to meet such a constraint because error limit values are encoded in the bitstream, and so the decompressor does not need to know how these values were selected.

3.2.4.2 Sample Representatives

The predictor cannot in general utilize the exact values of the original data samples because these values will not be available to the decompressor at the time of reconstruction when $m_z(t) > 0$. Consequently, a *sample representative* $s''_z(t)$ is used in place of the original sample value for the purpose of calculating subsequent predictions.

At the decompressor, assigning each reconstructed sample to be equal to the center of the quantizer bin, $s'_z(t)$, will minimize the maximum reconstruction error. However, this choice may not minimize other distortion metrics such as mean squared error (see 5.4 for an example).

Similarly, selecting the sample representative $s''_z(t)$ to be equal to $s'_z(t)$, which is the traditional predictive compression approach, does not always optimize compression performance, even when compression is lossless.

For this reason, user-specified parameters ϕ_z, ψ_z, and Θ are used to control the calculation of sample representatives. The sample representative $s''_z(t)$ is an integer approximation to

$$\frac{\phi_z}{2^\Theta}\hat{s}_z(t) + \left(1 - \frac{\phi_z}{2^\Theta}\right)\left(s'_z(t) - \frac{\psi_z}{2^\Theta}\,\mathrm{sgn}[q_z(t)]m_z(t)\right).$$

This is illustrated in figure 3-6 for $m_z(t) = 2$, $q_z(t) = 2$, and nonzero values of ϕ_z and ψ_z. The sample representative always lies in the range from quantizer bin center $s'_z(t)$ to the predicted sample value $\hat{s}_z(t)$; the values of ϕ_z, ψ_z, and Θ control the compromise between these two limits. Setting $\phi_z = \psi_z = 0$ causes the sample representative to be equal to $s'_z(t)$, and larger values of these parameters yield sample representative values closer to $\hat{s}_z(t)$. It should be noted that the sample representative may be outside of the quantizer bin containing $s'_z(t)$.

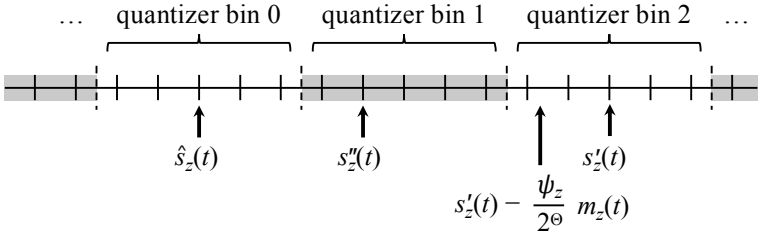


Figure 3-6: Sample Representative Calculation

3.2.5 MAPPED QUANTIZER INDICES

A quantizer index may be positive or negative, but the variable-length codes used by the entropy coder are defined only on nonnegative integer inputs. Thus each quantizer index $q_z(t)$ is mapped to a nonnegative integer $\delta_z(t)$, called the *mapped quantizer index*. This mapping is invertible, so that the decompressor can reconstruct the original quantizer index $q_z(t)$ 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 produce more output bits when given larger input values. So for effective compression, the mapping assigns smaller magnitude quantizer indices to smaller mapped values so that more accurate predictions are more effectively encoded.

The mapping used in the Recommended Standard is

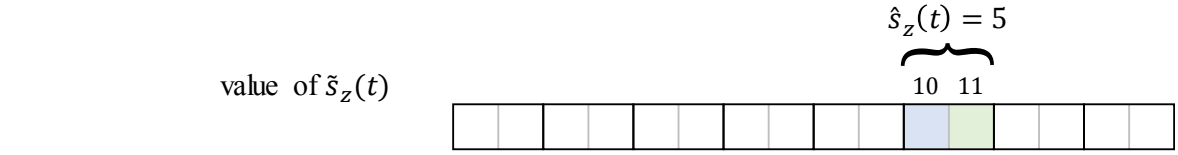
$$\delta_z(t) = \begin{cases} |q_z(t)| + \theta_z(t), & |q_z(t)| > \theta_z(t) \\ 2|q_z(t)|, & 0 \le (-1)^{\tilde{s}_z(t)} q_z(t) \le \theta_z(t), \\ 2|q_z(t)| - 1, & \text{otherwise} \end{cases} \tag{14}$$

where

$$\theta_z(t) = \begin{cases} \min\{\hat{s}_z(0) - s_{\min}, s_{\max} - \hat{s}_z(0)\} & t = 0 \\ \min\left\{\left\lfloor \dfrac{\hat{s}_z(t) - s_{\min} + m_z(t)}{2m_z(t) + 1} \right\rfloor, \left\lfloor \dfrac{s_{\max} - \hat{s}_z(t) + m_z(t)}{2m_z(t) + 1} \right\rfloor\right\} & t > 0. \end{cases}$$

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 quantizer indices.

As an example, figure 3-7 shows the values of the mapped quantizer indices when the predicted sample value is $\hat{s}_z(t) = 5$ for an image with unsigned 3-bit samples under lossless compression ($m_z(t) = 0$). The predicted sample value $\hat{s}_z(t) = 5$ could have been produced by $\tilde{s}_z(t) = 10$ (shown in blue) or $\tilde{s}_z(t) = 11$ (in green). The former case indicates a prediction that $q_z(t) < 0$ is more likely than $q_z(t) > 0$, and mapped quantizer indices are assigned accordingly.



value of $\tilde{s}_z(t)$						10	11	
value of $s_z(t)$	0	1	2	3	4	5	6	7
value of $q_z(t)$	−5	−4	−3	−2	−1	0	1	2
value of $\delta_z(t)$ (if $\tilde{s}_z(t) = 10$)	7	6	5	3	1	0	2	4
value of $\delta_z(t)$ (if $\tilde{s}_z(t) = 11$)	7	6	5	4	2	0	1	3

Figure 3-7: Example of Mapped Quantizer Index 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 quantizer index can be computed by inverting equation (14), which yields

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

Finally, from the value of $q_z(t)$ and $\hat{s}_z(t)$, the range of possible values for $s_z(t)$ is $[s'_z(t) - m_z(t), s'_z(t) + m_z(t)]$.

3.2.6 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
$s''_z(t)$	sample representative	$[s_{\min}, s_{\max}]$	D
$\tilde{s}''_z(t)$	double-resolution sample representative	$[2s_{\min}, 2s_{\max}+1]$	$D+1$
$s'_z(t)$	clipped quantizer bin center	$[s_{\min}, s_{\max}]$	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	$[-2^{\Omega+2}, 2^{\Omega+2} - 1]$	$\Omega+3$
$\check{s}_z(t)$	high-resolution predicted sample value	$[2^{\Omega+2}s_{\min}, 2^{\Omega+2}s_{\max} + 2^{\Omega+1}]$	$D+\Omega+2$
$\tilde{s}_z(t)$	double-resolution predicted sample value	$[2s_{\min}, 2s_{\max}+1]$	$D+1$

Symbol	Meaning	Bounds	Bits to Represent
$\hat{d}_z(t)$	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: $\Omega+3+\left\lceil \log_2\left[(4P+9)\left(2^D-1\right)\right]\right\rceil$ reduced mode: $\Omega+5+\left\lceil \log_2\left[P\left(2^D-1\right)\right]\right\rceil$
$\hat{s}_z(t)$	predicted sample value	$[s_{\min},s_{\max}]$	D
$e_z(t)$	double-resolution prediction error	$\left[-2^{D+1}+1, 2^{D+1}-2\right]$	$D+2$
$\Delta_z(t)$	prediction residual	$\pm\left(2^D-1\right)$	$D+1$
$q_z(t)$	quantizer index	$\pm\left(2^D-1\right)$	$D+1$
$\delta_z(t)$	mapped quantizer index	$\left[0, 2^D-1\right]$	D

Subsection 4.2.5 discusses the register size parameter R.

3.3 ENCODER

3.3.1 GENERAL

The encoder losslessly encodes the mapped quantizer indices 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 quantizer indices $\delta_{z,y,x}$ from the predictor. The body is produced by one of three entropy coding options: the *sample-adaptive* entropy coder, the *hybrid* entropy coder, or the *block-adaptive* entropy coder.

The original FL algorithm uses an adaptive coding approach using length-limited Golomb-Power-Of-2 (GPO2) codes (reference [15]), similar to the approach used in the JPEG-LS image compression standard (reference [16]). The sample-adaptive entropy coder specified in the Recommended Standard formalizes a version of this encoder. Under this entropy coding approach, each mapped quantizer index 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 hybrid entropy coder specified in the Recommended Standard is a modified version of the one originally used by the FLEX entropy coder. It includes codes equivalent to the length-limited GPO2 codes used by the sample-adaptive encoder, but augmented with an additional 16 variable-to-variable length ‘low-entropy’ codes. The hybrid entropy coder adaptively switches between these two coding methods on a sample-by-sample basis using code selection statistics similar to those used by the sample-adaptive coder. A single output

codeword from a low-entropy code may encode multiple samples, which allows the hybrid coder to obtain lower compressed data rates than can be produced by the sample-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-32 Recommended Standard (reference [17]). This encoding option 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 quantizer indices 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 quantizer indices 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.

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. 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 when the block-adaptive or sample adaptive coding options are used. When the hybrid coder is used, the end of the compressed image must be known to the decompressor.

3.3.2 HEADER

The header includes fields for all compression settings needed to reconstruct the compressed image. The Recommended Standard allows the following fields to be omitted from the header even when the values encoded in these fields would be needed by the decompressor:

a) weight initialization table, when custom weight initialization is used (see 5.3.3.3.2 of reference [1]);

b) weight exponent offset table, encoding the values of the weight exponent offsets in a case where they are not all set to zero (see 5.3.3.3.3 of reference [1]);

c) damping table subblock, encoding the values of the damping parameter ϕ_z in a case where not all ϕ_z are set to zero (see 5.3.3.5.2 of reference [1]);

d) offsets table subblock, encoding the values of the damping parameter ψ_z in a case where not all ψ_z are set to zero (see 5.3.3.5.3 of reference [1]);

e) when the sample-adaptive entropy coder is used, a user-defined accumulator initialization table.

Omitting such information might be a logical choice, for example, if a mission used the same fixed custom weight initialization values 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 SAMPLE PROCESSING ORDER

The mapped quantizer indices are sequentially processed by the entropy coder in the order selected by the user (and indicated in the header, see subsection 5.4.2 of reference [1]) and encoded in the compressed image body. The Recommended Standard specifies the allowed orders in which mapped quantizer indices may be input to the entropy coder; this order need not correspond to the order in which samples are output from the imaging instrument or processed by the predictor.

When the hybrid entropy coder is used, each output codeword encodes one or more mapped quantizer indices, but the output codeword that encodes quantizer index $\delta_z(t)$ may appear in the compressed image body *after* the output codeword that encodes $\delta_z(t + 1)$ (see 3.3.5). For this reason, it may be misleading to refer to the 'encoding order', so in this case, it is preferable to refer to the order in which samples are input to, or processed by, the entropy coder.

In addition to BSQ input order, the Recommended Standard also allows Band-Interleaved (BI) input order, which includes BIL and BIP as special cases. Under BI input order, a *frame*, defined as the set of all sample values with the same y coordinate (see subsection 3.2.3 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, processing proceeds in BIP order. Figure 3-8 illustrates the BI input order for samples in a frame when $M = 3$.

BIL and BIP input 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.

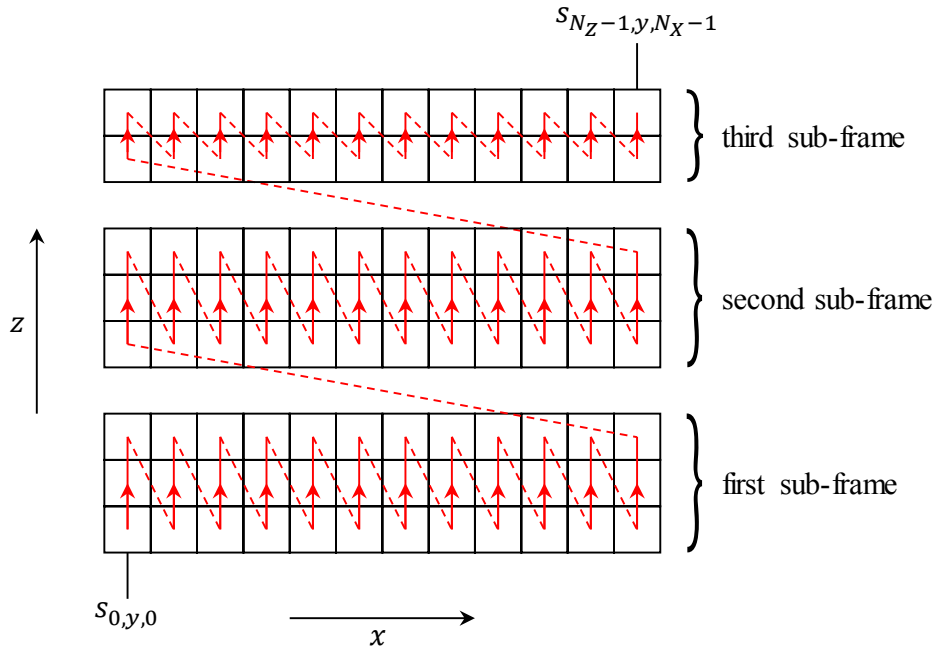


Figure 3-8: Illustration of Sample Processing Order within a Frame under Band Interleaved Order with Sub-Frame Interleaving Depth $M = 3$

Because the predictor adapts separately for each spectral band, all scan orders produce the same sample value predictions. Under the sample-adaptive and hybrid entropy coding options, separate entropy coding statistics are maintained for each spectral band; compressed image size is independent of sample input order for the sample-adaptive option and nearly so for the hybrid coder. However, as discussed in 4.3, under the block-adaptive entropy coding option, compressed image size will vary somewhat depending on the sample input 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 [15] and [18], 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 [16].

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

$$\delta = u \cdot 2^k + r, \tag{15}$$

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 quantizer index, the sample-adaptive encoder maintains a running sum $\Sigma_z(t)$ of mapped quantizer indices 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 quantizer index value, and this estimate is used to select the code parameter k. Specifically, k is the largest nonnegative integer $k \leq D-2$ satisfying

$$2^k \leq \frac{\Sigma_z(t)}{\Gamma(t)} + \frac{49}{128}, \tag{16}$$

and $k = D-2$ when this relation is not satisfied by any $k < D-2$ (see reference [19] for an analysis).

The counter and accumulator are each periodically halved (see subsection 5.4.3.2.3.4 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 (15) can be used to reconstruct the value of δ.

3.3.5 HYBRID ENTROPY CODER

The hybrid entropy coder includes codes equivalent to the length-limited GPO2 codes used by the sample-adaptive encoder, but augmented with an additional 16 variable-to-variable length 'low-entropy' codes to provide better compression of low-entropy data. Such low-entropy data become more prevalent as quantization step size increases.

3.3.5.1 Enabling Backwards Decoding

An interesting feature of the hybrid entropy coder specified in the Recommended Standard is that it is designed so that decoding proceeds in reverse order. This permits a simpler and more memory-efficient encoder implementation than FLEX's original hybrid entropy coder, which was based on an interleaved entropy coding approach (references [20] and [21]).

The encoding of each low-entropy mapped quantizer index makes use of one of 16 variable-to-variable length codes. A single output codeword from a low-entropy code may encode multiple mapped quantizer indices, which allows lower compressed data rates than can be achieved by the high-entropy codes. Each high-entropy mapped quantizer index immediately produces an output codeword that is written to the compressed bitstream, while each low-entropy code waits until enough data has arrived to determine the next output codeword. Consequently, the output codeword that encodes quantizer index $\delta_z(t)$ may appear in the compressed image body *after* the output codeword that encodes $\delta_z(t+1)$.

By decoding the compressed image body in reverse order, the decoder can accommodate the varying latency between the arrival of a low-entropy mapped quantizer index and its ultimate encoding. This is possible because (1) the output codewords from the high- and low-entropy codes are suffix-free (i.e., no codeword is a suffix of another codeword) rather than prefix-free; (2) the compressed image ends with a 'tail' that encodes the final state of each low-entropy code and the final high-resolution accumulator value for each band (see subsection 5.4.3.3.5.4 of reference [1]); and (3) immediately prior to each rescaling of the high-resolution accumulator, the high-resolution accumulator's least significant bit is output so that the decoder can invert this rescaling operation.

3.3.5.2 Code Selection Statistics

For each band z, the hybrid encoder maintains a *high-resolution accumulator* $\tilde{\Sigma}_z(t)$ and a *counter* $\Gamma(t)$. With each new mapped quantizer index $\delta_z(t)$, the high-resolution accumulator is incremented by $4\delta_z(t)$; the counter is incremented by one when t is incremented. Both the counter and accumulator are rescaled periodically at an interval controlled by the user-specified *rescaling counter size* parameter γ^*. These code selection statistics are similar to the corresponding ones for the sample-adaptive encoder, except that the accumulator for the hybrid coder has two bits higher resolution, and updates to code selection statistics are performed *before* encoding $\delta_z(t)$, because decoding proceeds in reverse order.

The ratio $\tilde{\Sigma}_z(t)/\Gamma(t)$ represents a scaled estimate of the mean mapped quantizer index for band z. This ratio determines the coding method used to encode $\delta_z(t)$. If this ratio exceeds a fixed threshold, $T_0/2^{14}$ (see subsection 5.4.3.3.5.1.4 of reference [1]), then $\delta_z(t)$ is said to correspond to a 'high-entropy' sample; otherwise, $\delta_z(t)$ is said to correspond to a 'low-entropy' sample. For both high- and low-entropy samples, the values of $\tilde{\Sigma}_z(t)$ and $\Gamma(t)$ also determine the specific code used to encode $\delta_z(t)$.

3.3.5.3 Encoding High-Entropy Mapped Quantizer Indices

Each high-entropy mapped quantizer index is encoded using a variable-length binary codeword from a family of codes. Each code in the family is equivalent to a length-limited GPO2 code, but with the output bits arranged in a different order so that the code is suffix-free.

The code parameter $k_z(t)$ used to encode a high-entropy mapped quantizer index $\delta_z(t)$ is determined by the values of $\tilde{\Sigma}_z(t)$ and $\Gamma(t)$, similar to the selection of the length-limited GPO2 code for the sample-adaptive coder.

In annex F it is proved that satisfying inequality (66) of reference [1] with $k_z(t) = 1$ could only occur for an input image with bit depth $D = 3$. For this reason, the Recommended Standard requires $k_z(t) \geq 2$, thus eliminating the need to implement the $k_z(t) = 1$ code, which would be used only in rare circumstances.

It should be noted that every codeword from the high-entropy code with parameter $k_z(t)$ has length at least $k_z(t) + 1$ bits, which is why the Recommended Standard requires $k_z(t) \leq D - 2$.

3.3.5.4 Encoding Low-Entropy Mapped Quantizer Indices

Each of the 16 low-entropy codes is a nonbinary-input, binary-output, variable-to-variable length code that defines a mapping from an exhaustive prefix-free set of variable-length *input codewords* over an input symbol alphabet that varies from code to code, onto an exhaustive suffix-free set of variable-length binary *output codewords*.

The ratio $\tilde{\Sigma}_z(t)/\Gamma(t)$ determines the code index *i* of the low-entropy code for $\delta_z(t)$.

Associated with each low-entropy code is an integer input symbol limit L_i, between 0 and 12 inclusive (see table 5-16 of reference [1]). One can think of L_i as distinguishing between 'likely' values ($\delta_z(t) \leq L_i$) and 'unlikely' values ($\delta_z(t) > L_i$).

If $\delta_z(t)$ is a likely value for the code ($\delta_z(t) \leq L_i$), then the component code encodes the value of $\delta_z(t)$. Otherwise, it encodes an 'escape' symbol, indicating that $\delta_z(t)$ was an unlikely symbol ($\delta_z(t) > L_i$) and the nonnegative residual value $\delta_z(t) - L_i - 1$ is encoded, using a code that is equivalent to a length-limited unary codeword, immediately preceding the output codeword from the low-entropy code. Since escape symbols occur with low probability, the

efficiency with which these residual values are encoded has only a small impact on overall coding effectiveness.

Input symbols assigned to a given low-entropy code are collected until a complete input codeword is formed for that code. The component codes are designed so that an escape symbol always causes the completion of an input codeword, and so at most, one residual value will accompany any output codeword from a low-entropy code.

The input symbol limit L_i limits the size of the input alphabet in the low-entropy codes by treating all unlikely symbols the same way. This permits a lower number of codewords in a component code. Table 3-2 shows the sizes of the component codes used.

Table 3-2: Entropy Code Sizes

Code Index	Input Symbol Limit	Number of Codewords	Maximum Input Length	Maximum Output Length (bits)
0	12	105	3	13
1	10	144	3	13
2	8	118	3	12
3	6	120	4	13
4	6	92	4	13
5	4	116	6	15
6	4	101	6	15
7	4	81	5	18
8	2	88	12	16
9	2	106	12	17
10	2	103	12	18
11	2	127	16	20
12	2	109	27	21
13	2	145	46	18
14	2	256	85	17
15	0	257	256	9

3.3.6 BLOCK-ADAPTIVE ENTROPY CODER

Reference [22] presents a thorough discussion of the CCSDS 121.0-B-3 Recommended Standard (reference [17]) used as the block-adaptive entropy coding option.

3.3.7 WORD SIZES

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

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

Symbol	Meaning	Bounds	Bits to Represent
$\Gamma(t)$	counter (sample-adaptive and hybrid entropy coders)	$[0,2^{\gamma^*}-1]$	γ^*
$\Sigma_z(t)$	accumulator (sample-adaptive entropy coder)	$[0,2^{D+\gamma^*}-1]$	$D+\gamma^*$
$\tilde{\Sigma}_z(t)$	high-resolution accumulator (hybrid entropy coder)	$[0,4(2^D-1)$ $(2^{\gamma^*}-1)]$	$D+\gamma^*+2$

3.4 BACKWARDS COMPATIBILITY AND LOSSLESS COMPRESSION

Issue 2 of the Recommended Standard (reference [1]) retains all features available in issue 1 (reference [8]), and the compressed image header of the Recommended Standard has been designed to be backwards compatible with issue 1. Thus compressed images produced by a compressor that is compliant with issue 1 will also be compliant with issue 2.

Table 2-1 in reference [1] enumerates features introduced in issue 2 that would need to be disabled to produce a compressor that is compliant with issue 1. It should be noted that features added in issue 2 are not limited to those that provide near-lossless compression capabilities. Thus, for example, a losslessly compressed image that is compliant with issue 2 might not be decompressible with a decompressor that is compliant with issue 1.

Even when compression is lossless, the use of new features (in particular, well-chosen values of sample representative offset ϕ_z and the use of the hybrid entropy coder) can sometimes provide improved compression performance over issue 1, as illustrated in table 3-4.

Table 3-4: Lossless Compression Data Rates for Issue 1 (Using the Sample-Adaptive Entropy Coder) and Issue 2 (Using the Hybrid Entropy Coder and Selecting Good Sample Representative Parameter Values)

Instrument	# bands	Data rate (bits/sample) Issue 1	Data rate (bits/sample) Issue 2	Δ rate (bits/ sample)
IASI	8461	4.75	4.75	0.00
AIRS	1501	4.30	4.22	0.08
CRISM FRT	545	5.06	4.89	0.17
CRISM HRL	545	4.57	4.40	0.17
CRISM MSP	74	2.55	2.51	0.04
AVIRIS-NG rad1	432	6.91	6.60	0.31
AVIRIS-NG rad2	432	5.36	5.06	0.31
AVIRIS-NG raw	432	5.12	4.86	0.26
M3 target	260	3.09	2.96	0.13
M3 global	86	2.14	2.17	-0.02
M3 rad1	85	6.60	6.55	0.05
M3 rad2	85	2.21	2.17	0.04
Hyperion flatfield	242	3.97	3.91	0.06
Hyperion raw	242	4.31	4.20	0.12
SFSI radiance	240	2.96	2.87	0.09
SFSI raw	240	4.67	4.48	0.19
AVIRIS_16 rad	224	3.74	3.70	0.04
AVIRIS_16 raw	224	5.98	5.95	0.02
AVIRIS_12 raw	224	2.68	2.62	0.06
HICO_128	128	5.10	5.00	0.11
HICO_87	87	4.29	4.27	0.01
CASI radiance	72	7.80	7.77	0.02
CASI raw	72	4.99	4.97	0.03
MODIS night	17	4.73	4.72	0.01
MODIS day	14	5.77	5.62	0.15
MODIS 500	5	7.20	7.17	0.03
MODIS 250	2	6.48	6.48	0.01
MSG	11	3.39	3.35	0.04
Landsat	6	3.37	3.35	0.02
Pléiades	4	7.11	7.10	0.01
VEGETATION	4	5.15	5.14	0.01
SPOT	3	4.53	4.52	0.00

Careful selection of compression options and parameters may allow significantly simpler implementation, though sometimes at the expense of compression performance. For example, an implementation that provides only lossless compression and constrains the sample representative offset ϕ_z to always be zero would eliminate both the quantization calculation ($\delta_z(t)$ would simply be equal to $\Delta_z(t)$) and the sample representative calculation ($s''_z(t)$ would simply be equal to $z_z(t)$). Subsection 5.8.3.2 includes additional discussion of implementation simplifications possible when only lossless compression is required.

3.5 DECOMPRESSION

When the decompressor produces reconstructed sample values using the quantizer bin center $s'_z(t)$, maximum absolute error of each reconstructed image sample is minimized, ensuring that $|s_z(t) - s'_z(t)| \leq m_z(t)$. This choice of reconstructed value is used for all examples in this Green Book, except where noted otherwise.

However, it should be noted that the Recommended Standard specifies the compressor but does not impose requirements on the decompressor. In particular, a different choice of reconstructed sample value may minimize other distortion metrics such as MSE. Subsection 5.4 includes an example.

4 COMPRESSION SETTINGS

4.1 INTRODUCTION

This section examines the influence of different compression settings on performance.

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

User-specified settings of absolute and/or relative error limits allow a user to limit the error in reconstructed image samples. These parameters can be adjusted to trade reconstructed image fidelity for compressed data rate.

For a given set of fidelity parameters, one would generally like to select the other compression parameters to minimize compressed data rate (discussed in this section) and/or reduce implementation complexity (see section 5).

The predictor parameters affect the predicted sample values, and thus changing predictor parameters will generally produce a different reconstructed image and a different compressed data rate. Mapped quantizer indices output from the predictor are losslessly encoded by the entropy coder, and so changing entropy coder parameters affects compressed data rate but does not affect the reconstructed image.

Experimental results, some of which were originally presented in references [23], [24], and [25], 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.

As illustrated in 3.4, making use of features introduced in issue 2, specifically sample representatives and the hybrid entropy coder, can improve compression effectiveness, although the resulting compressed image could not be decompressed by a decompressor that is compliant with issue 1 but not issue 2. Here, the terms *high-performance* and *backwards-compatible* are used to indicate whether these new features were used to produce a given set of results. In this section, results are produced using high-performance compression settings, except as noted otherwise.

4.2 PREDICTOR

4.2.1 LOCAL SUM TYPE, PREDICTION MODE, AND NUMBER OF PREDICTION BANDS

Issue 2 of the Recommended Standard defines two different prediction modes (*full* and *reduced*) and four different local sum types (*column-* and *neighbor-oriented*, each with a *wide* and a *narrow* option) 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 [26]). Such streaking-artifacts are often evident in raw images from pushbroom imagers (see 4.2.1.3).

The choice of local sum type, prediction mode, and number of prediction bands (P) determine the prediction neighborhood, that is, the neighboring samples that directly influence the prediction of a given image sample.

Different choices for these three parameters strongly influence implementation complexity due to the varying data dependencies of each mode. In particular, to facilitate pipelining in a hardware implementation, issue 2 introduces two new local sum options (*narrow* neighbor-oriented and *narrow* column-oriented). More details about these options are given in the next subsection.

Figure 4-1 reports results for all choices of local sum type and prediction mode under varying number of prediction bands and absolute error limit. As in issue 1, the choice of both local sum type and prediction mode can have a major impact on compression performance, and the optimum choice strongly depends on the image type. The relative performance of these choices generally follows what one would expect based on the presence or absence of streaking artifacts in the input image. The use of column-oriented local sums and reduced mode is recommended for images that exhibit significant streaking artifacts parallel to the y direction (e.g., AVIRIS-NG raw images). On the AIRS, HICO, and Landsat images, which do not exhibit streaking artifacts, neighbor-oriented local sums outperform column-oriented local sums and the choice of prediction mode does not always have a significant impact on performance.

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 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.

Varying the absolute error limit parameter appears to generally have little impact on which choice of local sum type and prediction mode is optimum. Results may vary at very high (in relation to image bit depth) quantization levels, as indicated for the Landsat images.

[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.

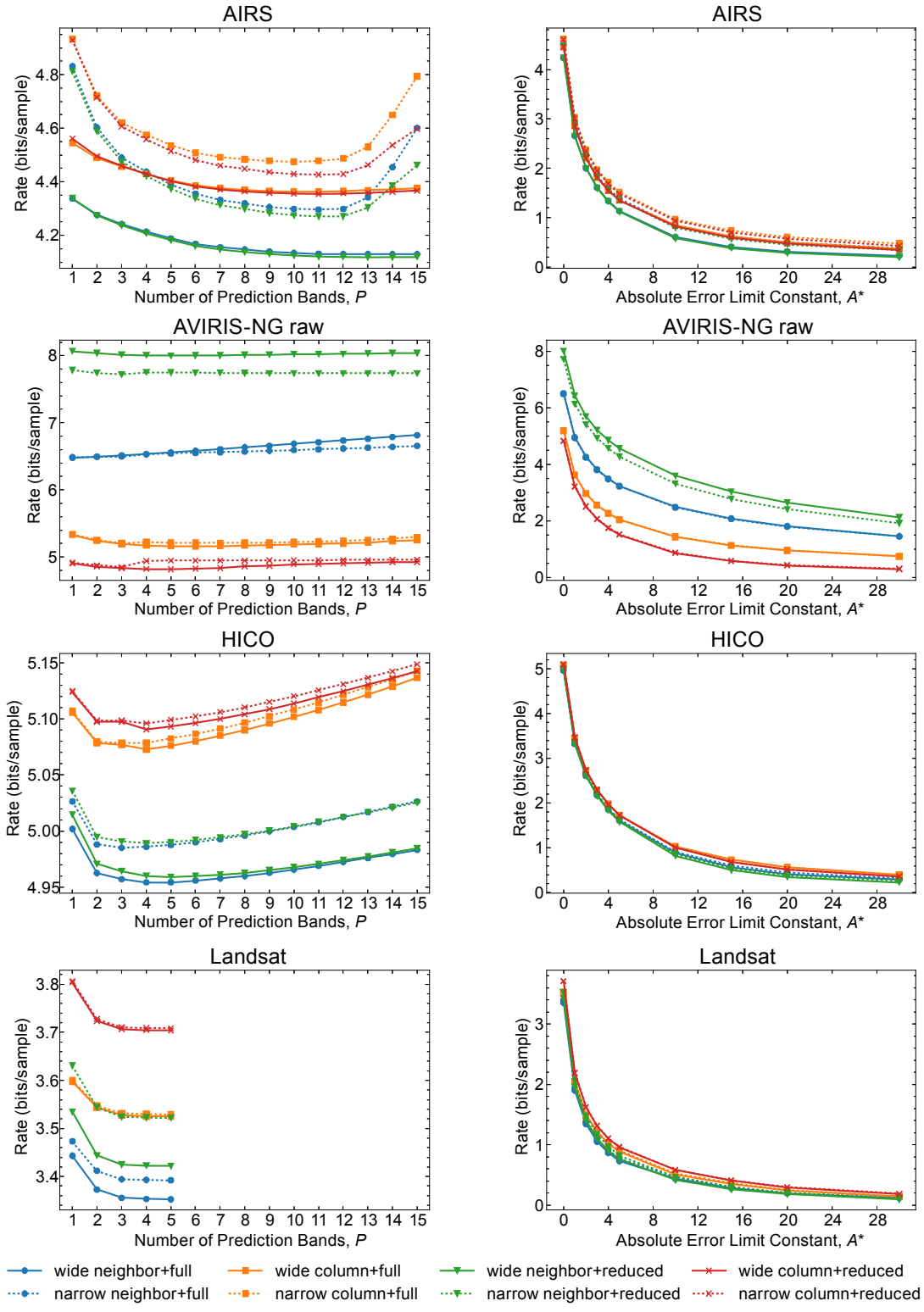


Figure 4-1: Compressed Data Rate for Different Choices of Prediction Mode and Local Sum Type

NOTE – Plots in the left column are for lossless compression ($A^* = 0$).

4.2.1.1 Local Sums: Narrow versus Wide

As discussed above, issue 2 defines four local sum options: the two local sum options in issue 1 are now called *wide neighbor-oriented* and *wide column-oriented*. The two new options, *narrow neighbor-oriented* and *narrow column-oriented* are defined as hardware-friendly alternatives. In particular, when combined with reduced prediction mode, the use of narrow local sums eliminates the dependency on sample representative $s''_{z,y,x-1}$ in the prediction calculation for neighboring sample $\hat{s}_{z,y,x}$. Eliminating this dependency may facilitate pipelining in a hardware implementation, though generally at the expense of somewhat reduced compression effectiveness.

It should be noted that the use of narrow local sums and full prediction mode might be an unlikely combination in practice since full prediction mode has data dependencies that may outweigh any advantage provided by narrow local sums. During the development of this standard, full prediction modes solving this data dependency (suppression of the 'W' directional local difference) were also considered, but extensive tests indicated no evidence of a performance gain over the use of the existing and less complex reduced mode.

In addition to figure 4-1 the following figures provide a few sample results with the objective of assessing the impact of the narrow local sums on compression efficiency. The results have been obtained setting the encoder to use $P = 3$ previous bands for prediction, and employing the sample-adaptive entropy coder. Both full and reduced modes have been employed.

Figure 4-2 compares the use of narrow versus wide neighbor-oriented local sums in full mode. As expected, the narrow mode entails some performance loss. Even though this penalty is generally low (hyperspectral images such as Landsat or AVIRIS images typically show a loss less than about 0.1 bit/sample), the loss can be relatively high in some cases, such as the AIRS sounder images.

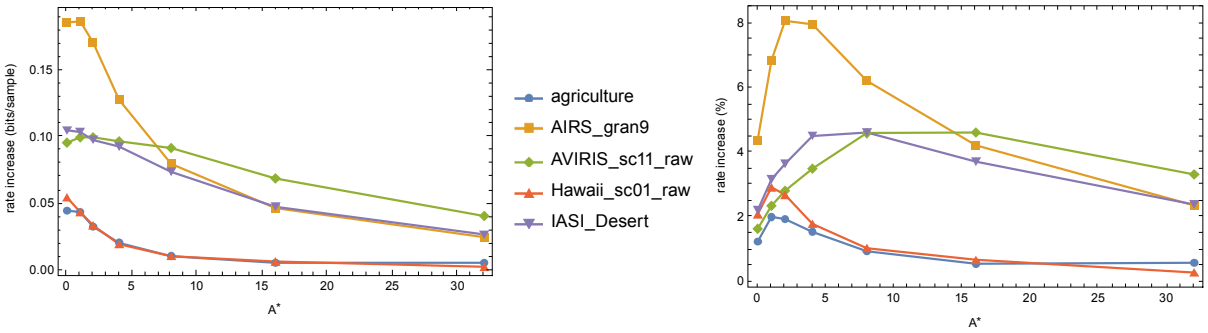


Figure 4-2: Rate Increase from Using Narrow Instead of Wide Neighbor-Oriented Local Sums in Full Mode

It is also interesting to compare the performance in reduced mode (see figure 4-3), which, in combination with narrow column-oriented local sums, enables low complexity and efficient pipelining in a hardware architecture by avoiding the use of the sample $s''_{z,y,x-1}$ in all steps of the prediction process. The performance cost of using narrow instead of wide column-oriented local sums is very small (typically around 0.02 bits/sample), making this solution very appealing for hardware implementations.

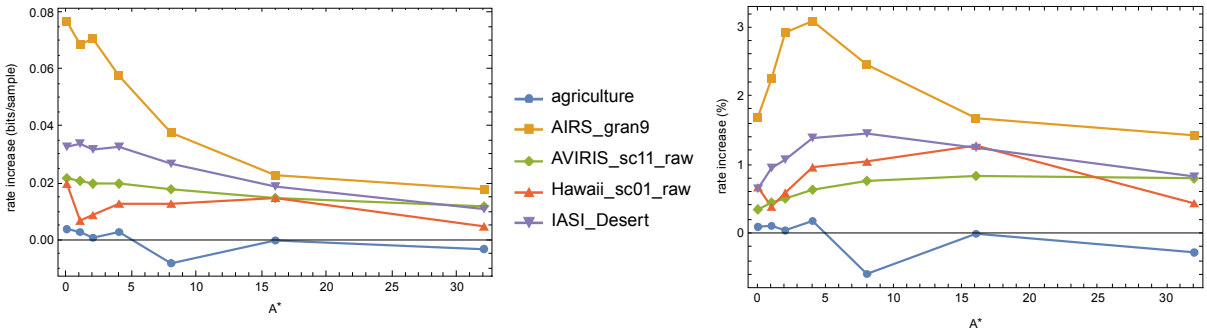


Figure 4-3: Rate Increase from Using Narrow Instead of Wide Neighbor-Oriented Local Sums in Reduced Mode

The results shown in the figures above and in the right column of figure 4-1 illustrate that the use of narrow local sums nearly always results in some performance penalty.[3] Narrow and wide column-oriented local sums differ only for the first image frame (i.e., at $y = 0$), and thus tend to yield only a small performance difference as long as image height is not small. For neighbor-oriented local sums, the penalty for using narrow instead of wide local sums becomes significant (10% or more in several cases) at larger values of absolute error.

4.2.1.2 Number of Bands for Prediction

Figures 4-1 and 4-2 show 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 some cases, visibly increase. In some cases (see figure 4-14), under specific compression settings (well-chosen sample representative parameters), smaller values of P can provide better performance (even for $P < 6$). For the images in the corpus, evidently there is not much motivation to use values of P near the maximum allowed, 15. A value of $P = 3$ appears to be a reasonable default for both lossless and near-lossless compression.

[3] The exception here is the use of neighbor-oriented local sums on AVIRIS-NG images. But in this case, the lower complexity column-oriented local sums perform substantially better anyway.

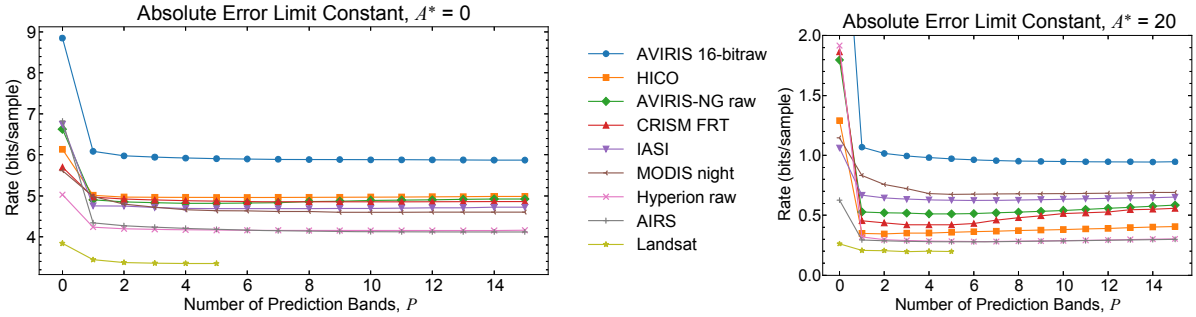


Figure 4-4: Average Compressed Bit Rate as a Function of *P*

4.2.1.3 Streaking Artifacts

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 4-5 shows the mean Digital Number (DN) value of each detector element of the Moon Mineralogy Mapper (M3) hyperspectral imager, averaged over several imaging frames. 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 4-6.

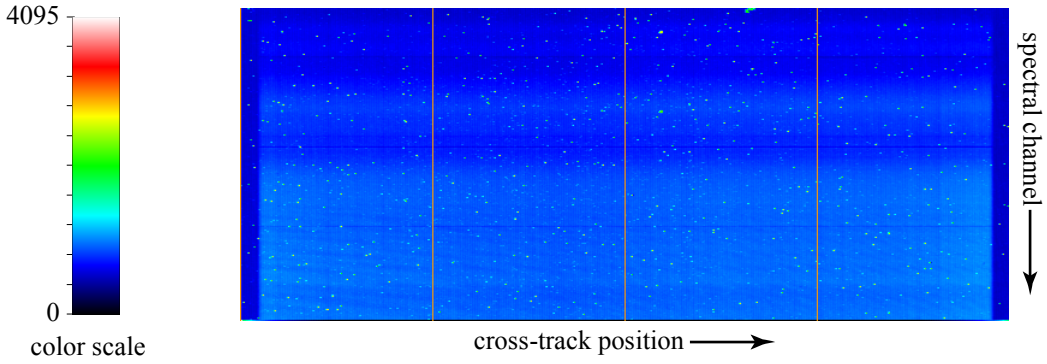


Figure 4-5: Mean DN Value of M3 Detector Array, Using Color Scale at Left, Averaged over 7000 Imaging Frames

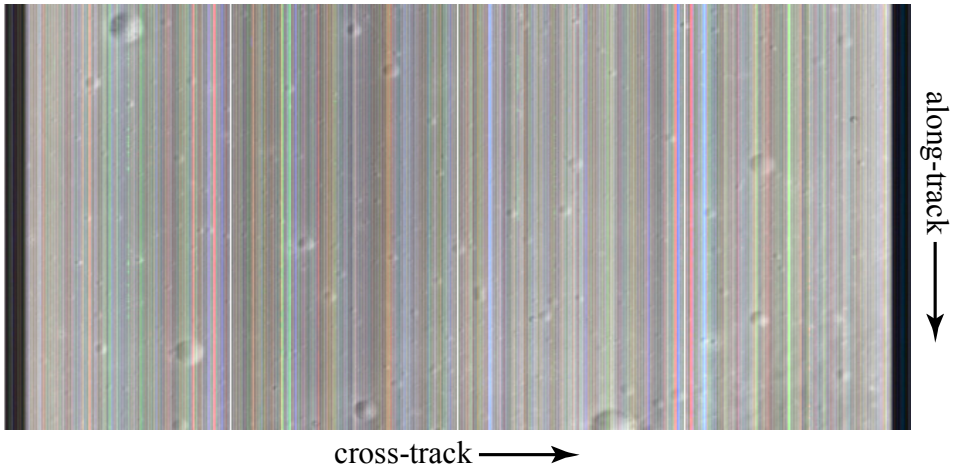


Figure 4-6: False-Color Image Derived from Spectral Channels 200, 201, 202 from a Portion of an M3 Image

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 5.4).

4.2.2 ADAPTATION RATE AND SAMPLE REPRESENTATIVES

The rate at which the predictor adapts to changing image characteristics is controlled by $v_{\min}$ and t_{inc} initially, and by $v_{\max}$ at steady state. Figure 4-7 shows the performance as a function of steady-state learning rate, $v_{\max}$, at different quantization levels. The results indicate that adequate values for $v_{\max}$ are generally invariant to the quantization level, though a more pronounced decrease in performance is evident as $v_{\max}$ becomes smaller.

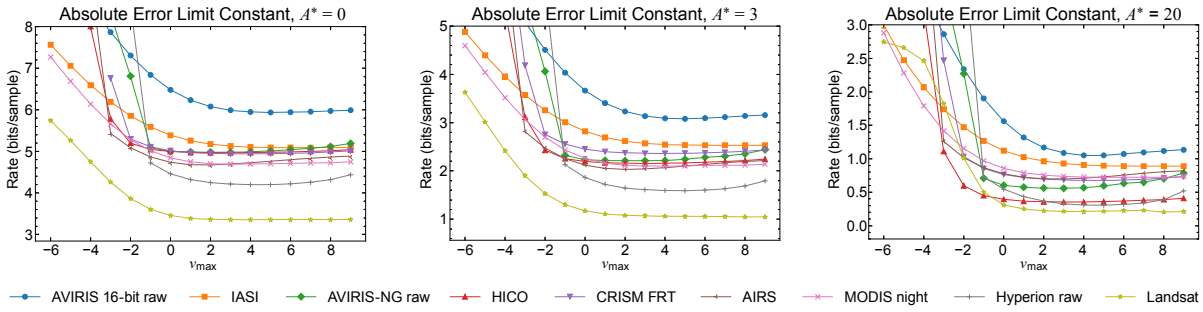


Figure 4-7: Average Compressed Data Rate as a Function of $v_{\max}$ When $v_{\min} = -6$ and $t_{\text{inc}} = 2^7$

Figure 4-8 shows the performance impact of extreme choices for $v_{\min}$ and t_{inc}. In images with few samples per band, parameters affecting the initial learning rate control the predictor learning rate for most (or all) samples in each band and thus have a significant impact on overall prediction performance. This effect diminishes as spatial size increases.

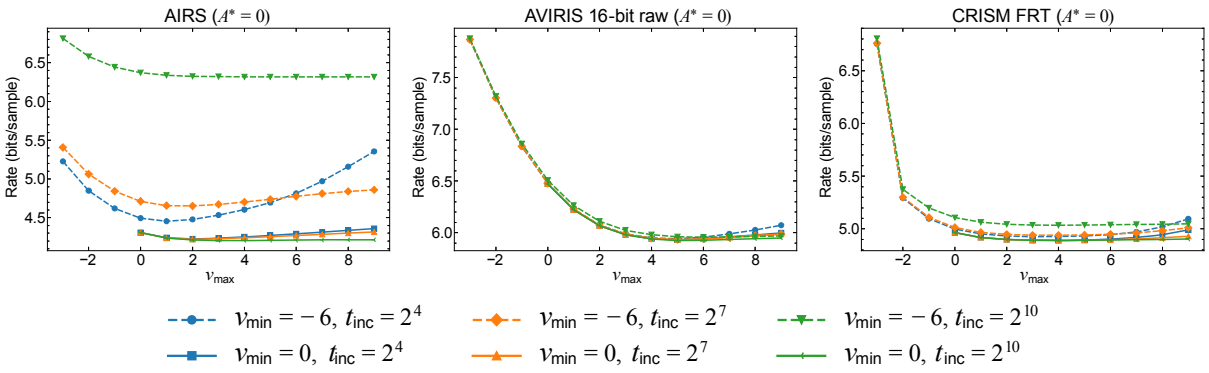


Figure 4-8: Average Compressed Data Rate for Different Choices of Parameters That Affect Predictor Adaptation

The sample representatives and the associated *damping* parameter ϕ_z interact significantly with the predictor learning rate. The damping parameter interacts with the predictor feedback loop in a way that may reduce the influence of image noise on the adaptation process, and thus may improve coding performance for noisy images. Figure 4-9 shows coding performance in relation to $v_{\max}$ at different damping values. Substantial benefits may be obtained from well-chosen damping values. For the test images, using sample representative resolution $\Theta = 3$, best results have been obtained by employing a strong damping value of 5 for AVIRIS-NG, HICO, SFSI; a medium value of 3 for AIRS, AVIRIS 12-bit raw, CASI, CRISM, Hyperion, and M3 Target; and no damping ($\phi_z = 0$) for the remaining images. Visual inspection of image bands seems to corroborate that noise levels are a determining factor in selecting the optimum damping value. However, these results do not exclude the possibility that other factors might also be relevant to the selection of damping values, such as instrument resolution or other forms of signal distortion different from noise.

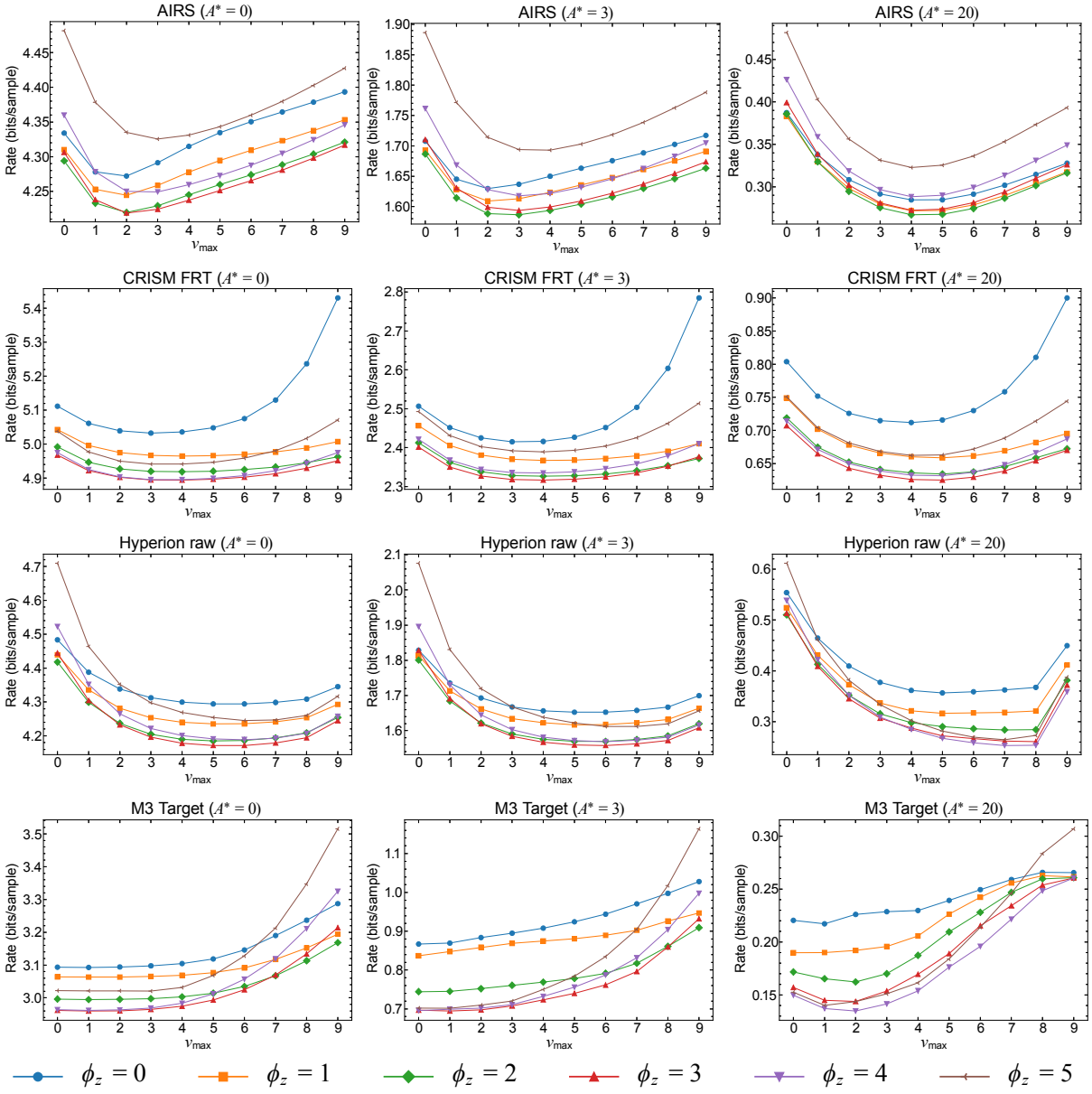


Figure 4-9: Average Compressed Data Rate as a Function of v_{max} for Different Values of Sample Representative Damping Value, ϕ_z, and Absolute Error Limit Constant A^*

Damping values are examined in relation to v_{min} and v_{max} in figures 4-9 and 4-10. Larger damping values have been found to coincide with decreased performance at higher v_{max} values. Setting $v_{max} = 7 - 2^{3-\Theta}\phi_z$ has been found a good choice in experimental results. Similarly, higher damping values seem to produce decreased performance when used with smaller v_{min} values. Thus higher damping values seem to narrow the desirable range of predictor learning rates.

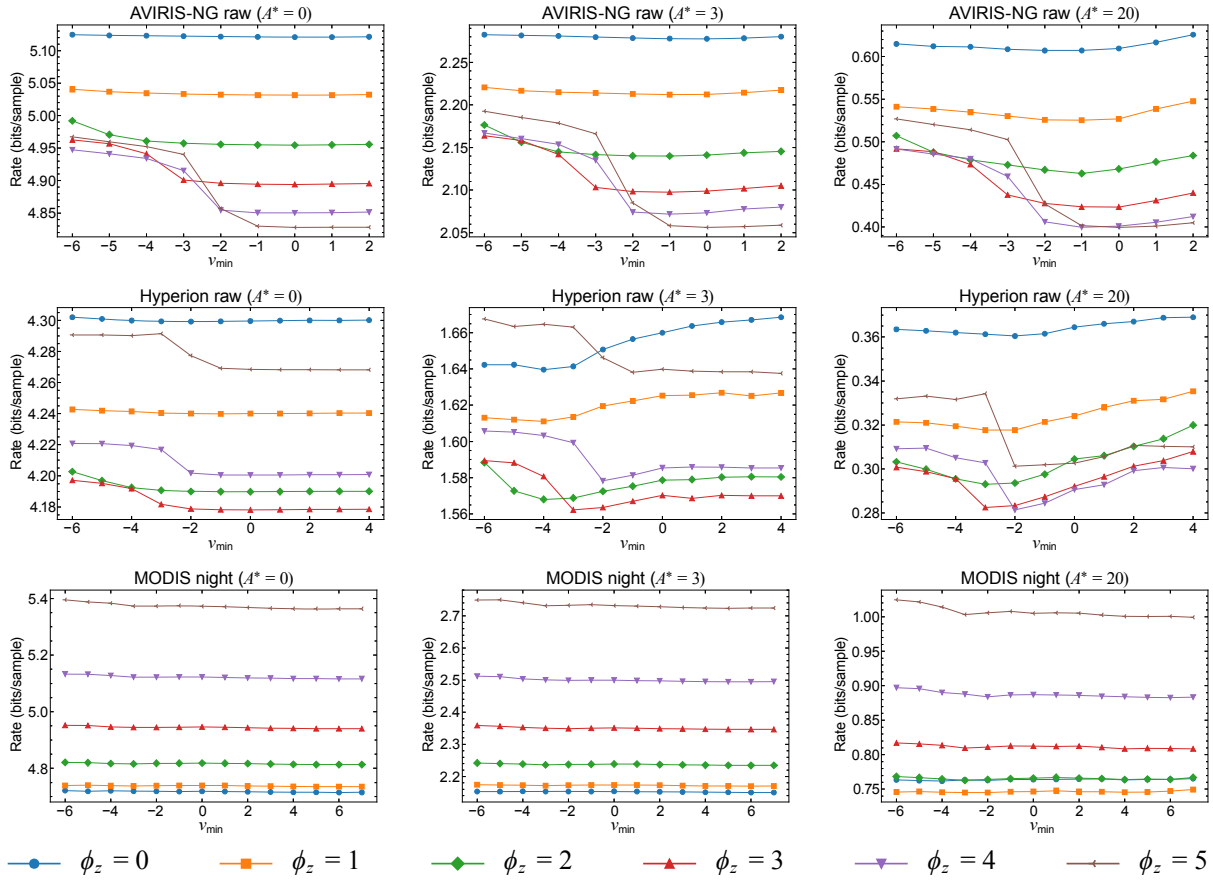


Figure 4-10: Average Compressed Data Rate as a Function of v_{min}, at Different Values of Sample Representative Damping, ϕ_z, and Absolute Error Limit Constant A^*

Figure 4-11 shows the relationship between damping value ϕ_z and the quantity of samples necessary to reach a steady predictor state, as controlled by t_{inc}. For the v_{min} and v_{max} values set as described above, varying ϕ_z seems to have little influence on optimizing the value of t_{inc}.

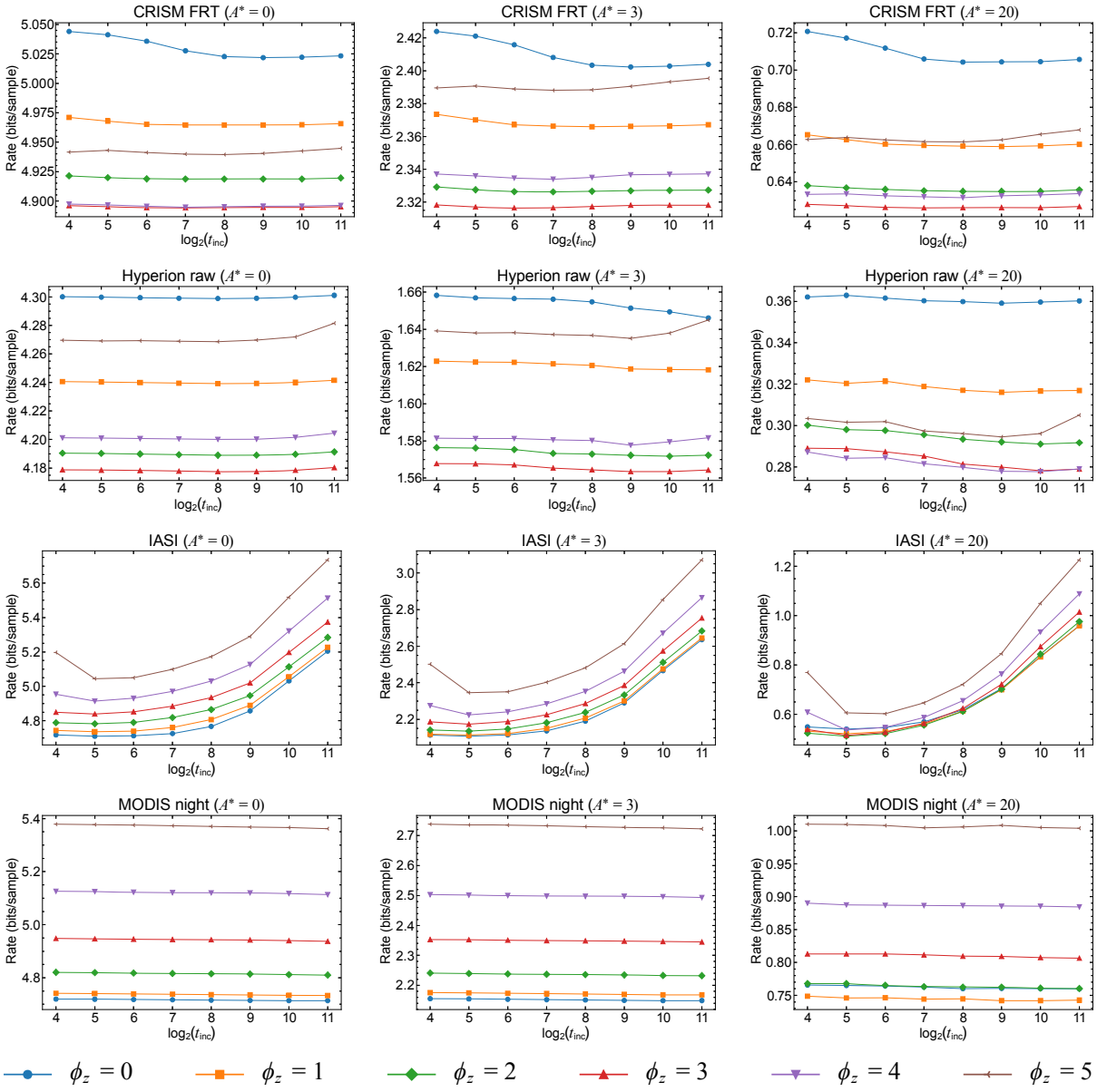


Figure 4-11: Average Compressed Data Rate as a Function of t_{inc} at Different Values of Sample Representative Damping, ϕ_z, and Absolute Error Limit Constant A^*

The *offset* parameter, ψ_z, also affects sample representative values, though this value has no effect when compression is lossless. For properly-selected values of ϕ_z, experimental results suggest that setting $\psi_z = 2^{\Theta} - 1$ is a reasonable default choice (figure 4-12).

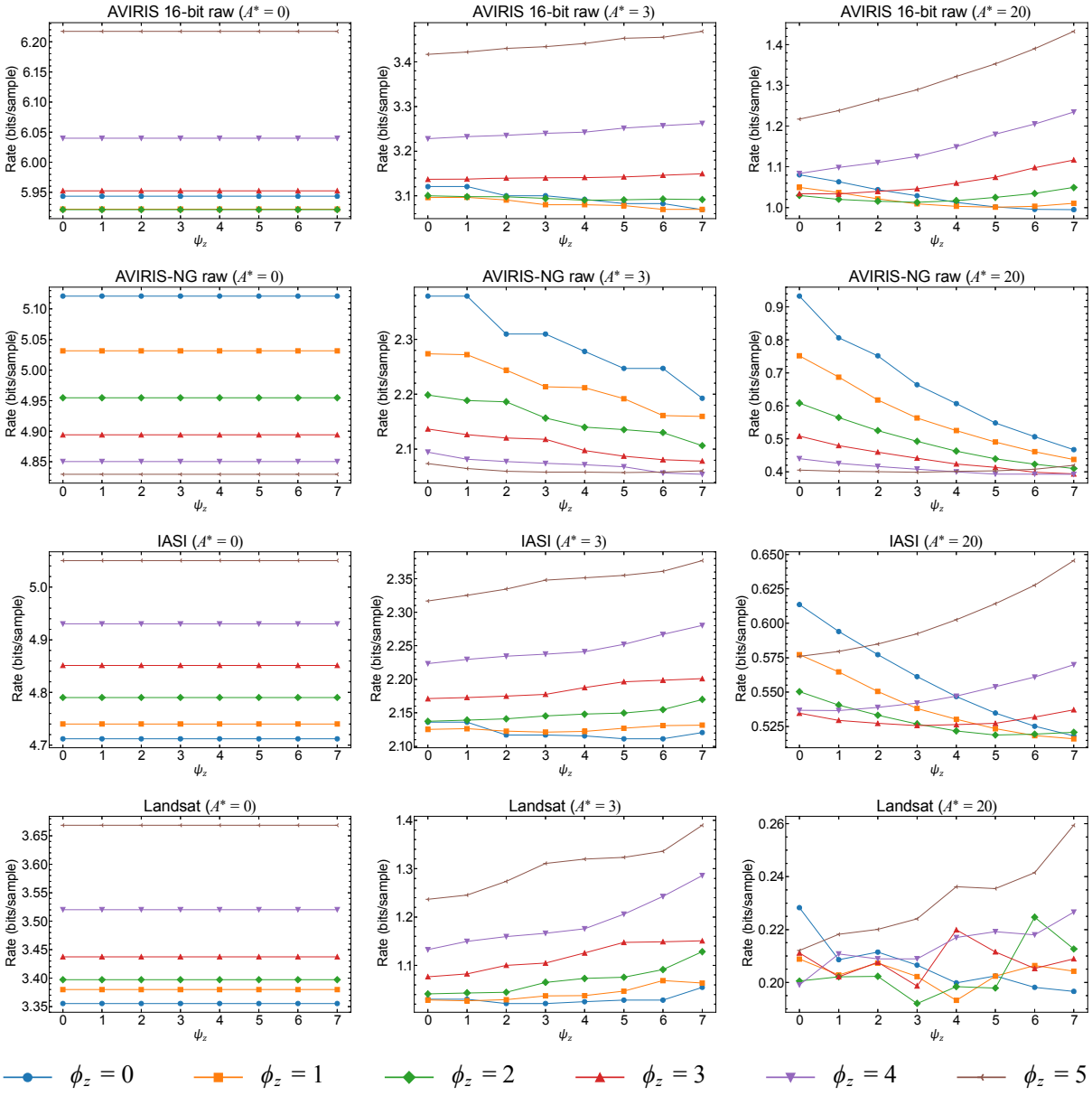


Figure 4-12: Average Compressed Data Rate as a Function of ψ_z at Different Values of Sample Representative Damping ϕ_z and Absolute Error Limit Constant A^*

To examine the relationship between sample representative parameters and instrument characteristics such as noise level and wavelength spacing, a processed test image was constructed. A raw AVIRIS-NG image (5nm band spacing) was partitioned into 2×8 blocks (2 samples in the cross-track direction and 8 samples in the along-track direction) in each band. Then the sample values in each block were added to produce a single sample, thus yielding a lower resolution image. Flat-field correction was applied to remove streaking artifacts caused by detector nonuniformity, and the resulting image was scaled so that all samples could be stored as 16-bit unsigned integers after rounding. This produced an image with high SNR.

From this high SNR image, 64 bands at 10 nm spacing (every other band) starting at 1558 nm were taken to produce an image with 300 cross-track samples, 512 frames, and 64 spectral bands. To observe how sample representative behavior changes with noise level, Gaussian distributed random noise with standard deviation σ was added. For several values of σ, figure 4-13 shows the relative increase in compressed image size (compared to using the optimum pair of sample representative parameters) as sample representative parameters varied using $A^* = 40$, $P = 2$, $\Theta = 4$, and $\Omega = 19$. The same pair of fixed values of (ϕ', ψ') were used in all bands except for the first, where $\phi'_1 = \psi'_1 = 0$. Here normalized offset and damping are defined as $\phi'_z = \phi_z/2^\Theta$ $\psi'_z = \psi_z/2^\Theta$, respectively. The figure below shows that as noise increases, better bit rate performance is obtained via larger values of ϕ'_z and ψ'_z for the test image.

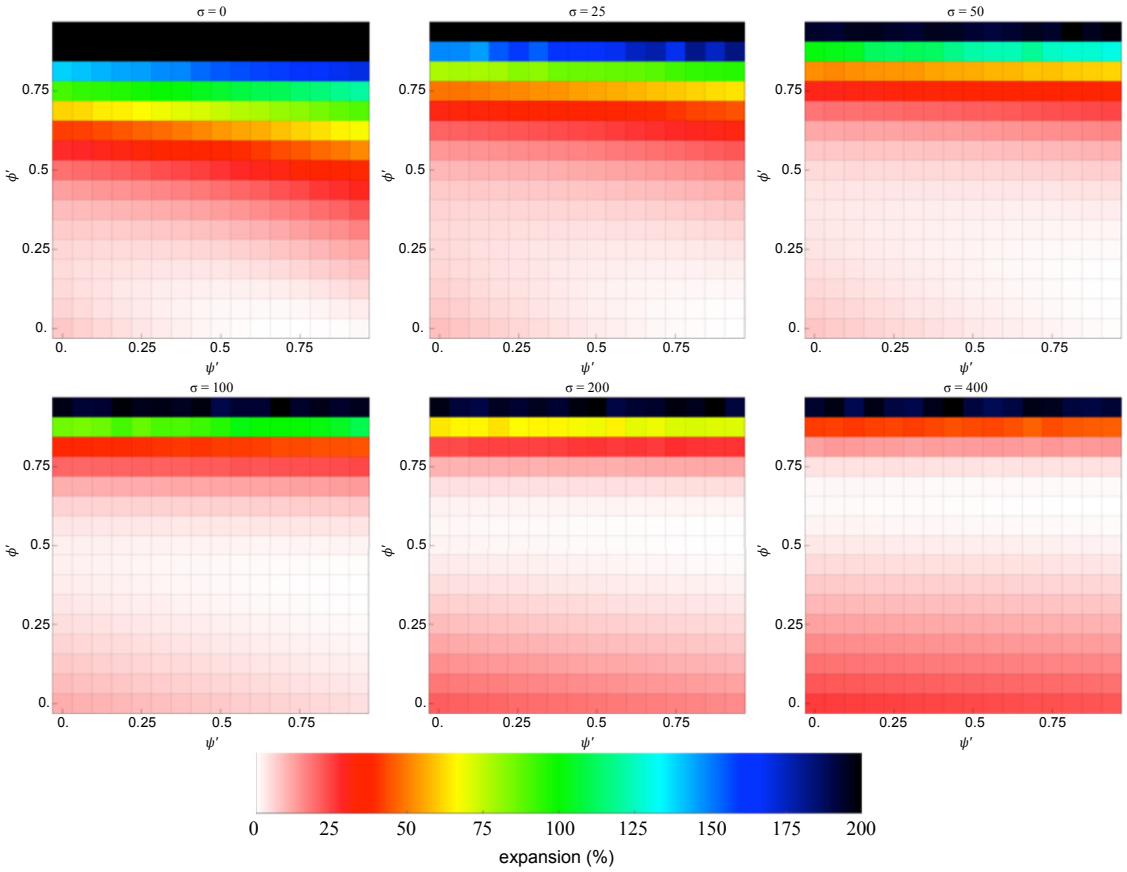


Figure 4-13: Increase in Compressed Image Size, Compared to Optimum Choice of (ϕ', ψ') at Different Noise Levels

One would expect prediction accuracy to improve when consecutive bands in an image are more similar, as might occur when band wavelengths are more closely spaced, and in this case, well-chosen values of ϕ_z and ψ_z may provide more of an improvement over the alternative of simply setting $\phi_z = \psi_z = 0$.

As an experiment, images formed by taking bands from the processed AVIRIS-NG image described above at different wavelength spacing are compressed. Specifically, 384 consecutive bands from 411 to 2330 nm from the image are selected. For a given wavelength spacing, the 384 bands were separated into groups to form 16 different images, each with 24 bands, with successive bands in each image having wavelengths that differ by the wavelength spacing value. For each of these 24-band images, an ad hoc algorithm was applied to find good $\{\phi'_z\}$ and $\{\psi'_z\}$ profiles, at different values of P, comparing the compressed bit rate (averaged over all 24 images) to that obtained by setting $\psi'_z = \phi'_z = 0$ for all z. In each case $v_{\min} = -1$ and $v_{\max} = 3$. Figure 4-14 shows the results when $A^* = 50$ and $\sigma = 100$.

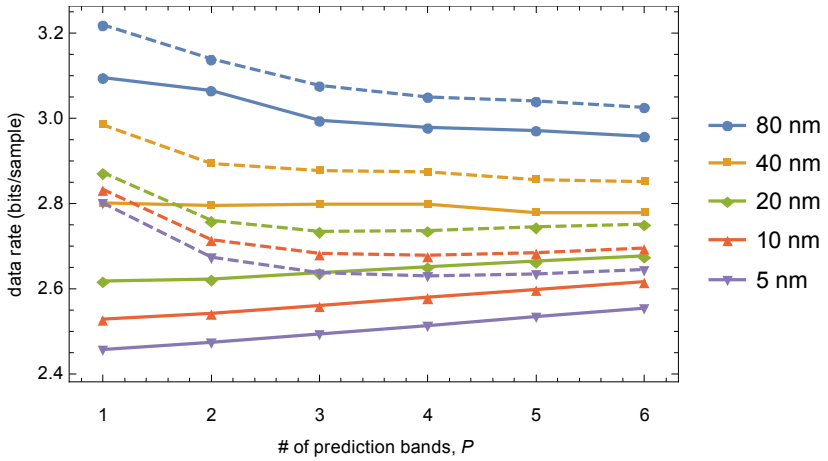


Figure 4-14: Compressed Data Rate Using Good $\{\phi'_z\}$ and $\{\psi'_z\}$ Profiles (Solid) and $\psi'_z = \phi'_z = 0$ (Dashed) as a Function of Number of Prediction Bands, P, at Different Wavelength Spacings

In this experiment, it is observed from figure 4-14 that the use of well-chosen sample representative parameters provides more benefit when the wavelength spacing is smaller. It also suggests that at smaller wavelength spacing, smaller values of P yield better performance provided that good sample representative parameters are selected.

Figure 4-15 examines compressed data rate performance as a function of sample representative parameters at different wavelength spacings. In this experiment, larger values of (ϕ', ψ') tend to give better performance when the wavelength spacing is smaller.

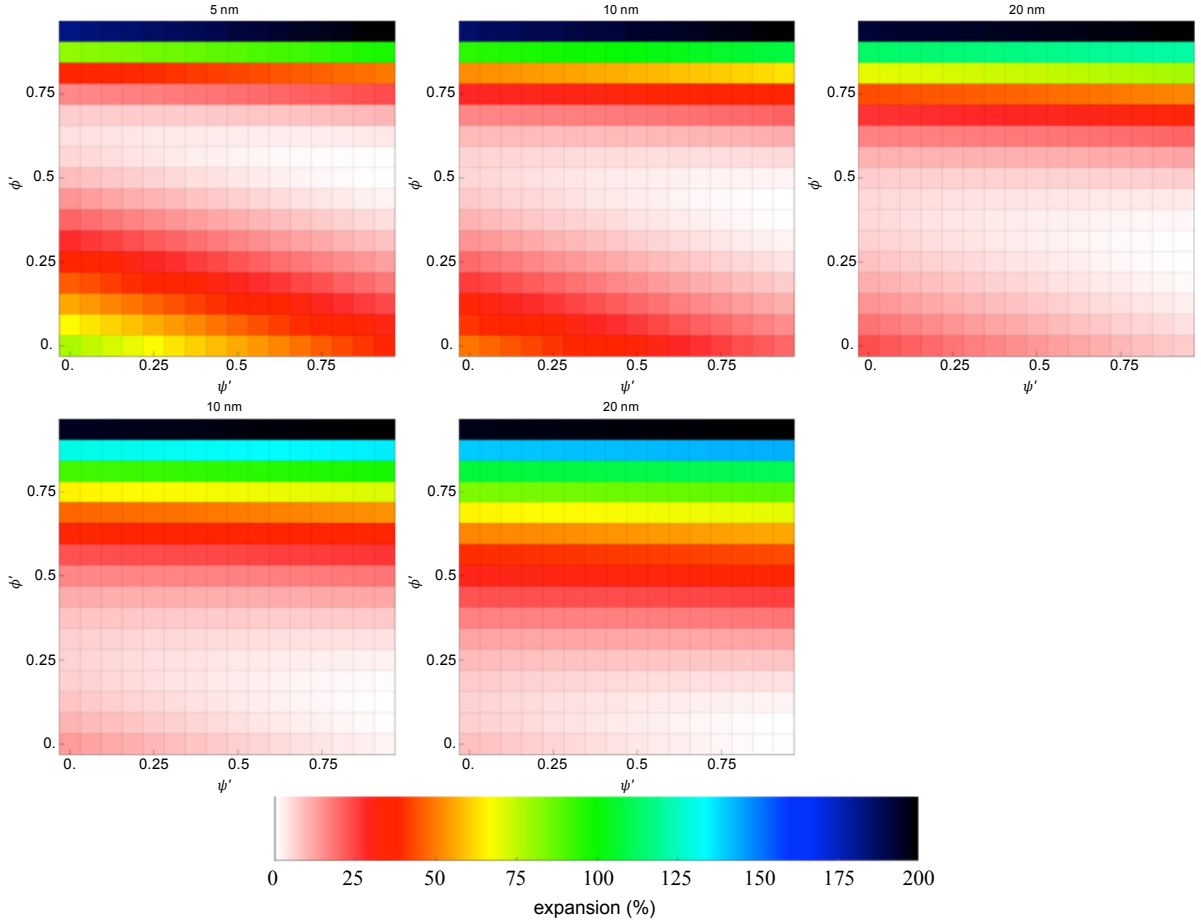


Figure 4-15: Average Expansion for $P = 1$ and $\sigma = 100$ at Different Band Spacings

NOTE – The same pair of fixed values of (ϕ', ψ') are used in all bands except for the first, where $\phi_1' = \psi_1' = 0$.

Varying (ϕ_z, ψ_z) while keeping error limit values fixed can affect both the compressed data rate and the reconstructed image. The maximum absolute error in the reconstructed image does not change (assuming that samples are reconstructed using quantizer bin center $s_z'(t)$), but the reconstructed image does change due to changing prediction accuracy, and other distortion metrics such as MSE may change. As an example, figure 4-16 illustrates a case where the error histogram in one band of an image changes noticeably when the sample representative parameter values are changed.

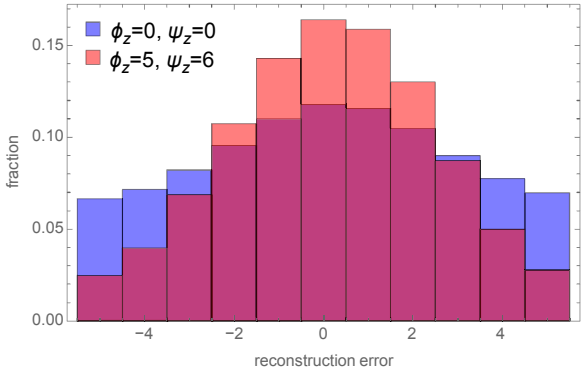


Figure 4-16: Histograms of Prediction Error in Band 169 of AVIRIS Hawaii Image Using Two Different Choices of (ϕ_z, ψ_z) with Θ = 4

4.2.3 WEIGHT EXPONENT OFFSETS

The use of weight exponent offsets $\left\{\zeta_z^{(i)}, \zeta^*\right\}$ allows fine tuning of the predictor learning rate for each band, and within a band for each predictor input as defined by prediction mode and number of prediction bands. Hence, weight exponent offsets can be seen as an extension to the learning rate control provided through $v_{\min}$ and $v_{\max}$.

A brute force approach has been followed to understand the potential gains that could yield well-informed choices of weight exponent offsets for some of the corpus images. The procedure employed is as follows. First, optimal values for $v_{\min}$ and $v_{\max}$ are found through exhaustive search. Then, all choices of weight exponent offsets are tested for the first image band. The best choice is kept for that band, and the same procedure is repeated for each remaining band, one by one, until all weight exponent offsets are set.

Results are reported in table 4-1. When a fixed damping value and no error limits are used for an entire image, as is the case here, the potential coding performance gains obtained by weight exponent offsets are scant.

Table 4-1: Data Reductions Obtained for Weight Exponent Offsets Set through Exhaustive Search

Image	Data Rate Reduction (%)
AVIRIS Yellowstone Scene 0 raw	0.3 %
Landsat Agriculture	0.2 %
MODIS 500m A2001123.0000	0.5 %
Pléiades Montpellier Misreg0	0.06%
Vegetation 1 1b	0.3 %

Nonetheless, given that good choices for v_{max} vary from instrument to instrument, a user may want to adjust the predictor learning rate for each image band when different per-band noise profiles suggest the use of different per-band sample representative damping values, or for images with large signal energy variation among bands. A data rate reduction of 7% was observed in a synthetic image produced from a high SNR image by simulating 32 consecutive very dark bands followed by 32 consecutive very bright bands when the synthetic image was encoded with band-dependent absolute error limits so that both dark and bright bands were encoded at roughly 1.5 bits/sample.

4.2.4 WEIGHT RESOLUTION

The precision with which the predictor stores its weight vector (i.e., its internal state) is controlled by the *weight component resolution* parameter, Ω, which 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.

Together with the *register size* parameter, R, both parameters determine the bit depths required for the multipliers employed in the predictor calculation. For each multiplier, the value of Ω controls the depth of one of its inputs (the other is determined by the image bit depth), while the value of R enables the multipliers to provide results modulo R, thus limiting its output bit depth.

The effects of varying weight resolution are reported in figures 4-17 and 4-18. As expected, when weight resolution is decreased, prediction accuracy is adversely affected by less accurate weight vector components, and thus rate is increased. For lossless compression, Figure 4-15 shows the relationship between weight resolution and rate, both in absolute and relative terms. For the images tested, weight resolution can be decreased to 11 bits with a rate increase less than 5%. For near-lossless compression, figure 4-18 shows the relationship between weight resolution and rate increase over the rate obtained with Ω = 19. It can be observed that the small perturbations in the curves for lossless compression ($A^* = 0$) are significantly amplified as A^* increases. While the predictor is approximately linear, lower values of Ω seem to magnify the non-linear effects that finite-precision operations have on the least significant bits.

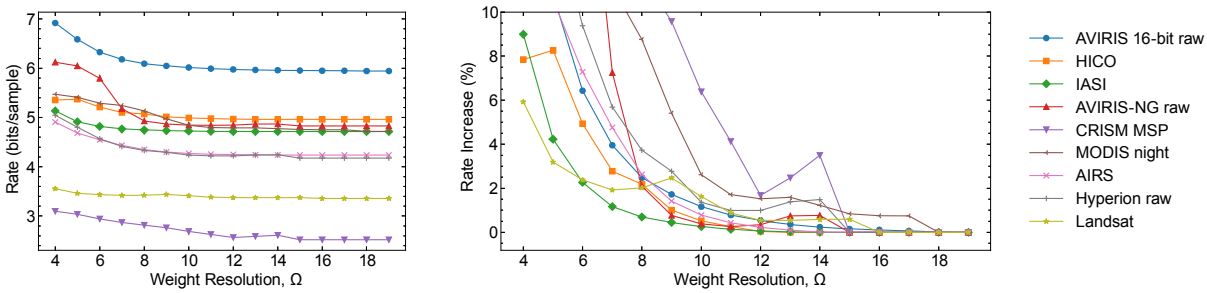


Figure 4-17: Average Data Rate as a Function of Weight Resolution, Ω, for Lossless Compression, Using $v_{max} = 3$

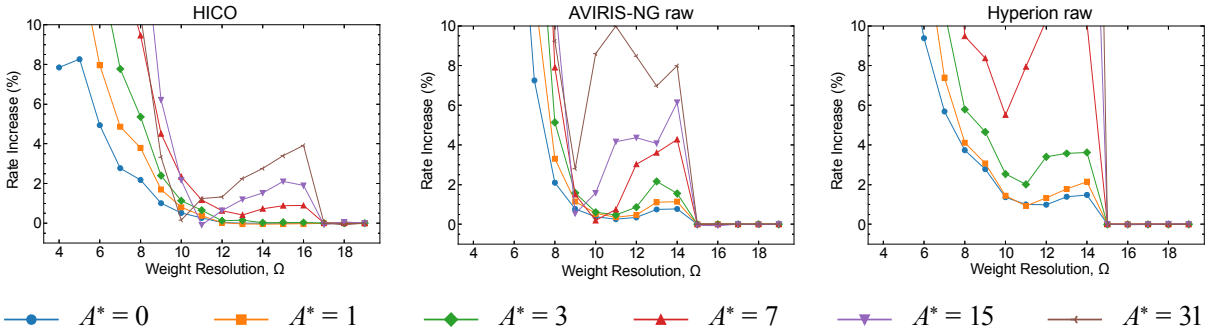


Figure 4-18: Change in Compressed Data Rate as a Function of Weight Resolution Ω, Relative to Ω = 19, at Different Values of Absolute Error Limit Constant A^*

Using values of Ω near the minimum allowed can significantly hurt compression effectiveness, but increasing Ω beyond a certain point may have little or no impact. In particular, in the weight update procedure, subsection 4.10 of reference [1], the amount by which a weight value is incremented is

$$\left\lfloor \frac{1}{2}\left(u \cdot \mathrm{sgn}^{+}[e_z(t)] \cdot 2^{-(\rho(t)+\varsigma)} + 1\right)\right\rfloor = \left\lfloor u \cdot \mathrm{sgn}^{+}[e_z(t)] \cdot 2^{-(\rho(t)+\varsigma+1)} + \frac{1}{2}\right\rfloor, \tag{17}$$

where u is some integer local difference value and ς is a weight exponent offset. 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

$$\left\lfloor \pm u \cdot 2^{\Omega-(v_{\max}+D+\varsigma+1)} + \frac{1}{2}\right\rfloor. \tag{18}$$

If $\Omega \geq v_{\max} + D + 2 + \varsigma$, 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; that is, the available weight resolution is not being fully used. Consequently, a reasonable rule-of-thumb is that using a weight resolution value larger than

$$\Omega^* = v_{\max} + D + 1 + \varsigma \tag{19}$$

is unlikely to improve compression effectiveness.

4.2.5 REGISTER SIZE

The main prediction calculation, subsection 4.7 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, the decompressor is able to reconstruct the sample value with at most $m_z(t)$ units of error, 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, all of the entropy coding options 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

$$\hat{d}_z(t) + 2^{\Omega}\left(\sigma_z(t) - 4s_{\text{mid}}\right) \tag{20}$$

cannot be represented as a signed R-bit quantity. Applying the bounds from table 3-1 in 3.2.6 to each term in the above expression, the range of possible values for this quantity is bounded by

$$\left[-2^{\Omega+1}\left(\left(2^D-1\right)(8P+\kappa)+1\right), 2^{\Omega+1}\left(\left(2^D-1\right)(8P+\kappa)-1\right)\right], \tag{21}$$

where the constant κ is

$$\kappa = \begin{cases} 1, & \text{reduced mode} \\ 19, & \text{full mode} \end{cases}. \tag{22}$$

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

$$R^* = \Omega + 2 + \left\lceil \log_2\left(\left(2^D-1\right)(8P+\kappa)+1\right)\right\rceil. \tag{23}$$

Thus if the register size R satisfies $R \geq R^*$, then overflow cannot occur in the prediction calculation. Table 4-2 tabulates the quantity R^* at maximum weight resolution, $\Omega = 19$. The table indicates that a 61-bit register is sufficient to guarantee that overflow will not occur in prediction for all possible choices of compression settings. The Recommended Standard requires $R \geq 32$, and so for some combinations of D, P, and Ω, overflow is impossible for any compliant implementation.

Table 4-2: 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
D=17	38	42	43	43	44	44	44	44	45	45	45	45	45	45	45	45
D=18	39	43	44	44	45	45	45	45	46	46	46	46	46	46	46	46
D=19	40	44	45	45	46	46	46	46	47	47	47	47	47	47	47	47
D=20	41	45	46	46	47	47	47	47	48	48	48	48	48	48	48	48
D=21	42	46	47	47	48	48	48	48	49	49	49	49	49	49	49	49
D=22	43	47	48	48	49	49	49	49	50	50	50	50	50	50	50	50
D=23	44	48	49	49	50	50	50	50	51	51	51	51	51	51	51	51
D=24	45	49	50	50	51	51	51	51	52	52	52	52	52	52	52	52
D=25	46	50	51	51	52	52	52	52	53	53	53	53	53	53	53	53
D=26	47	51	52	52	53	53	53	53	54	54	54	54	54	54	54	54
D=27	48	52	53	53	54	54	54	54	55	55	55	55	55	55	55	55
D=28	49	53	54	54	55	55	55	55	56	56	56	56	56	56	56	56
D=29	50	54	55	55	56	56	56	56	57	57	57	57	57	57	57	57
D=30	51	55	56	56	57	57	57	57	58	58	58	58	58	58	58	58
D=31	52	56	57	57	58	58	58	58	59	59	59	59	59	59	59	59
D=32	53	57	58	58	59	59	59	59	60	60	60	60	60	60	60	60

	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
D=17	43	43	44	44	44	44	45	45	45	45	45	45	45	45	46	46
D=18	44	44	45	45	45	45	46	46	46	46	46	46	46	46	47	47
D=19	45	45	46	46	46	46	47	47	47	47	47	47	47	47	48	48
D=20	46	46	47	47	47	47	48	48	48	48	48	48	48	48	49	49
D=21	47	47	48	48	48	48	49	49	49	49	49	49	49	49	50	50
D=22	48	48	49	49	49	49	50	50	50	50	50	50	50	50	51	51
D=23	49	49	50	50	50	50	51	51	51	51	51	51	51	51	52	52
D=24	50	50	51	51	51	51	52	52	52	52	52	52	52	52	53	53
D=25	51	51	52	52	52	52	53	53	53	53	53	53	53	53	54	54
D=26	52	52	53	53	53	53	54	54	54	54	54	54	54	54	55	55
D=27	53	53	54	54	54	54	55	55	55	55	55	55	55	55	56	56
D=28	54	54	55	55	55	55	56	56	56	56	56	56	56	56	57	57
D=29	55	55	56	56	56	56	57	57	57	57	57	57	57	57	58	58
D=30	56	56	57	57	57	57	58	58	58	58	58	58	58	58	59	59
D=31	57	57	58	58	58	58	59	59	59	59	59	59	59	59	60	60
D=32	58	58	59	59	59	59	60	60	60	60	60	60	60	60	61	61

The effects of varying the register size parameter, R, are illustrated in figure 4-19, with the region $R \geq R^*$ indicated via shading. Even when overflow is mathematically possible, it may be very rare. Decreasing register size by one or two bits less than R^* does not increase rate significantly, while decreasing by a few more bits produces large rate increases.

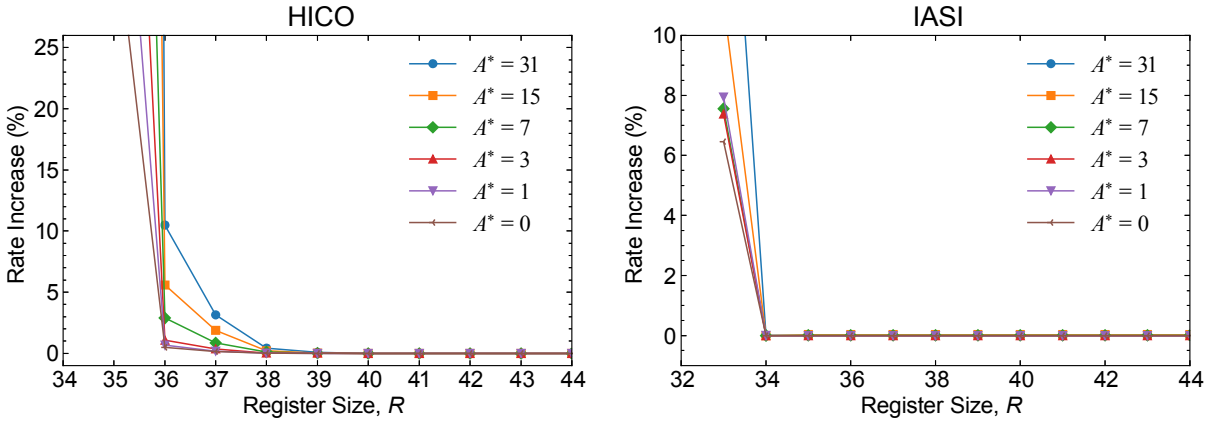


Figure 4-19: Change in Compressed Data Rate as a Function of Register Size, *R*, at Several Values of Absolute Error Limit Constant A^*

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, for example, 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 4-20).

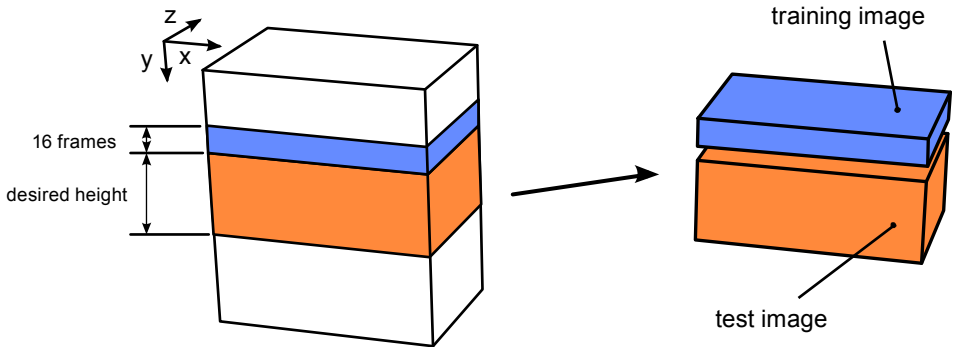


Figure 4-20: Training and Test Images Employed to Assess the Impact of Custom Weight Initialization

The experiment is as follows. First, the training image is losslessly compressed using default weight initialization. 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 losslessly compressed twice; once using default weight initialization, and once using custom weights obtained from the preserved weight vectors from the previous compression. This allows measurement of the impact of custom weight initialization. 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. The sample-adaptive entropy coder was used in each case.

Figure 4-21 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 $Q = \lfloor \Omega / 2 \rfloor$ for lossless compression. Multiple heights are simulated for the test 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 4-22 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 4-23 shows the average improvement as a function of $v_{\min}$ when $Q = \lfloor \Omega / 2 \rfloor$ at image heights of 16 and 32.

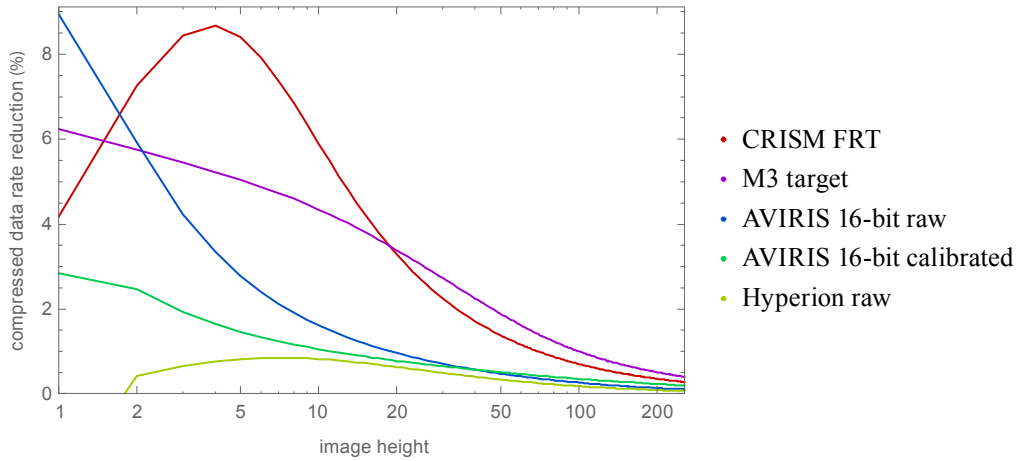


Figure 4-21: Average Reduction in Compressed Data Rate Obtained by Using Custom Weight Initialization (with $v_{min} = 3$, $Q = \lfloor \Omega / 2 \rfloor$) and Backwards-Compatible Compression Settings, Compared to Default Weight Initialization, as a Function of Image Height

NOTE – Larger values are better.

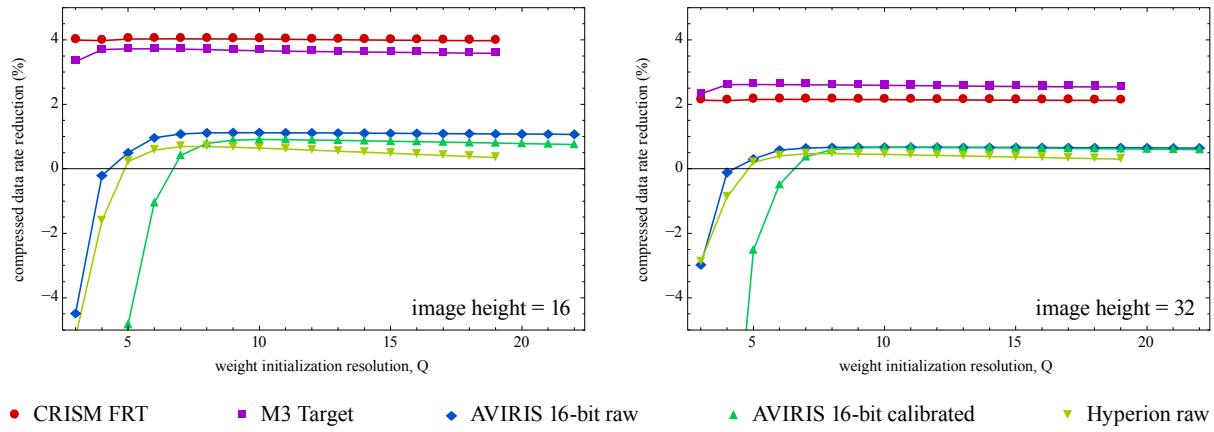


Figure 4-22: Average Reduction in Compressed Data Rate Obtained by Using Custom Weight Initialization (with $v_{min} = 3$, for Image Heights 16 and 32) and Backwards-Compatible Compression Settings, Compared to Default Weight Initialization, as a Function of Weight Initialization Resolution, *Q*

NOTE – Larger values are better.

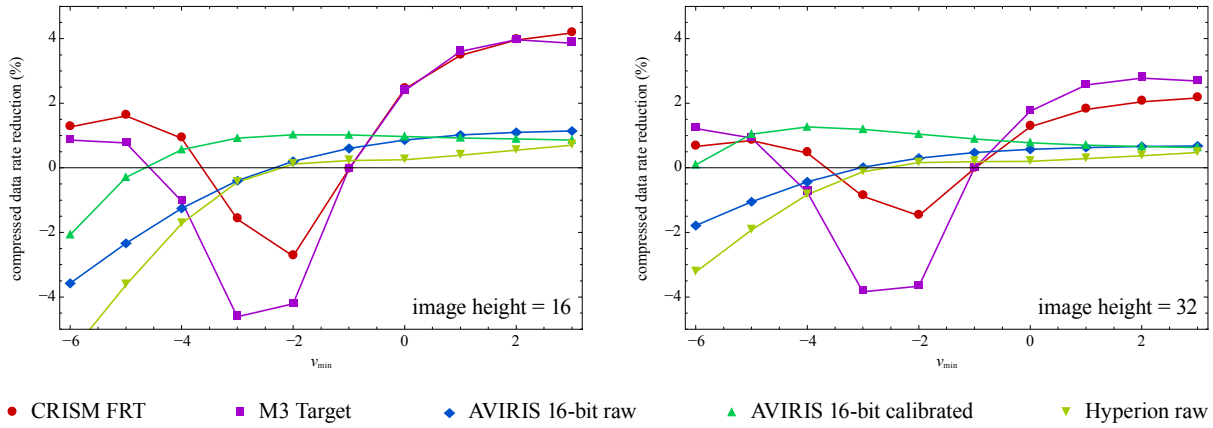


Figure 4-23: Average Reduction in Compressed Data Rate Obtained by Using Custom Weight Initialization (with $Q=\lfloor \Omega/2 \rfloor$, for Image Heights 16 and 32) and Backwards-Compatible Compression Settings, Compared to Default Weight Initialization, as a Function of v_{min}

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 samples. 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.

4.2.7 QUANTIZATION

4.2.7.1 Absolute and Relative Error Limits

The use of absolute and/or relative error limits allows a user to control reconstructed image fidelity. As an illustrative example, consider the AVIRIS Maine image (shown in figure 4-24), which includes both bright regions (clouds) and dark regions (cloud shadows and a lake).

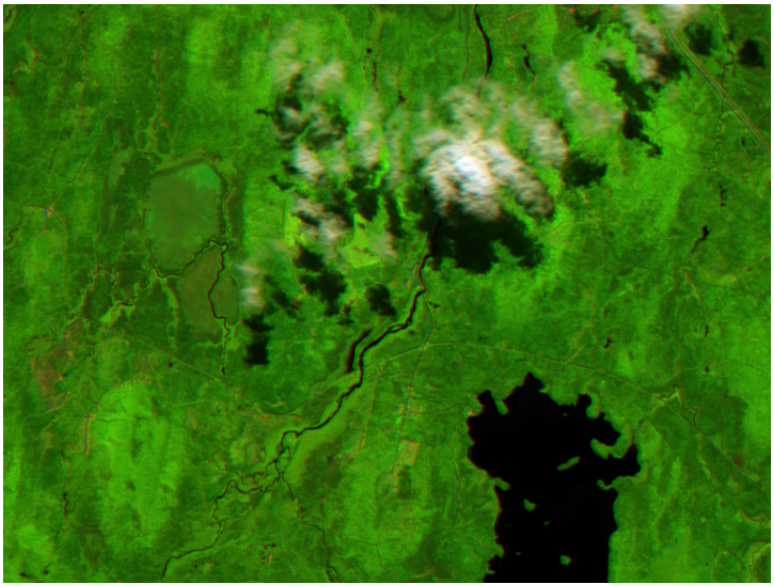


Figure 4-24: False-Color Image Derived from Bands 125, 70, 30 of 12-bit AVIRIS Maine Image

The image is compressed in three different ways, each producing bit rate of approximately 0.5 bits/sample: (a) using absolute error control, with absolute error limit constant $A^* = 4$, (b) using relative error control, with relative error limit constant $R^* = 72$, and (c) using both absolute and relative error control, with $A^* = 6$ and $R^* = 85$. Reconstructed sample values are produced using the quantizer bin center $s'_z(t)$.

Figure 4-25 provides a visualization of absolute error magnitude in three bands of the image reconstructed under the three different fidelity control settings, figure 4-26 shows the histograms of absolute error in those three bands, figure 4-27 shows relative error magnitude in the three bands, and figure 4-28 shows the corresponding histograms of relative error.

As can be seen, controlling fidelity using absolute error limits produces a nearly uniformly distributed reconstruction error, whereas using relative error limits clearly yields smaller error in the lake and cloud shadow regions, with higher error in other regions of the image, especially in the bright clouds. Controlling fidelity via relative error limits tends to yield more uniformly distributed relative error magnitudes.

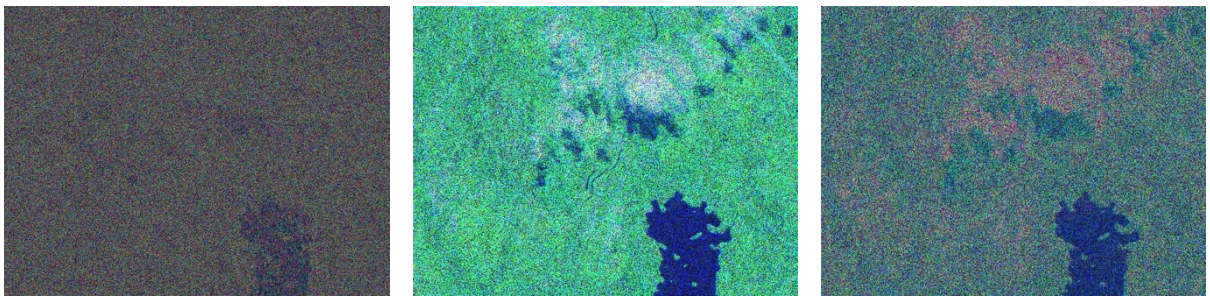


Figure 4-25: Absolute Error Magnitude $|s_z(t) - s'_z(t)|$ for (a) Absolute Error Control, (b) Relative Error Control, and (c) Both Absolute and Relative Error Control

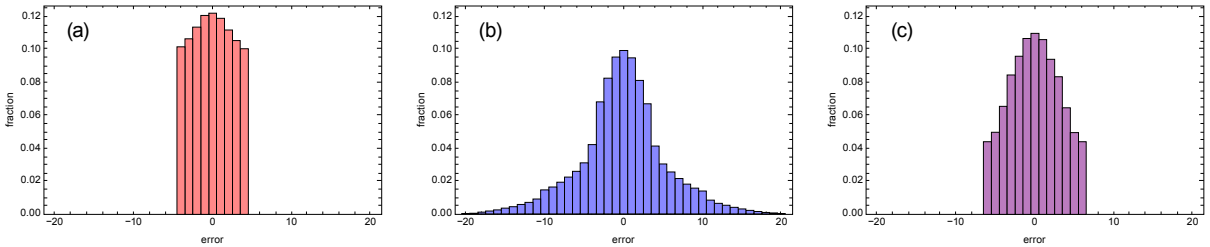


Figure 4-26: Histograms of Reconstruction Error $|s_z(t) - s'_z(t)|$ for (a) Absolute Error Control, (b) Relative Error Control, and (c) Both Absolute and Relative Error Control

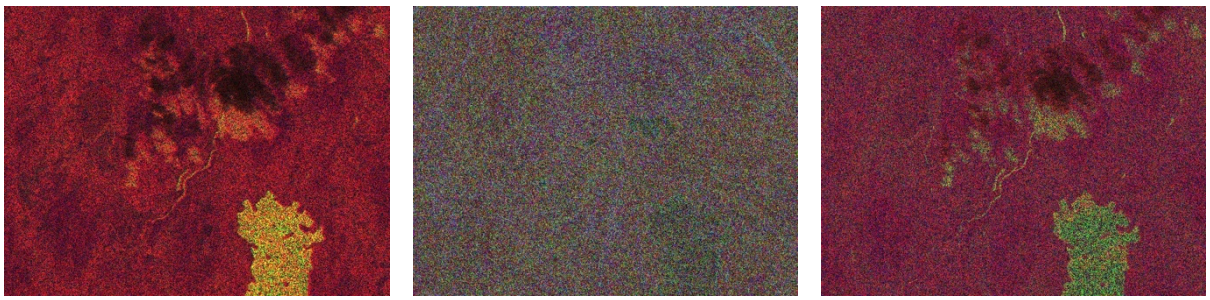


Figure 4-27: Relative Error Magnitude $s_z(t) - s'_z(t) / s_z(t)$ for (a) Absolute Error Control, (b) Relative Error Control, and (c) both Absolute and Relative Error Control

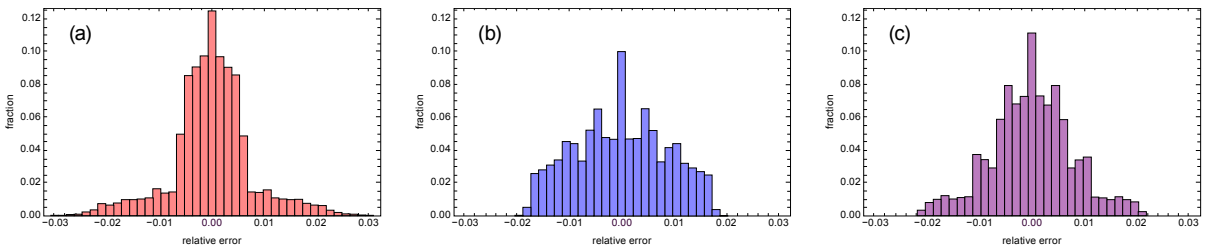


Figure 4-28: Histograms of Relative Reconstruction Error $s_z(t) - s'_z(t) / s_z(t)$ for (a) Absolute Error Control, (b) Relative Error Control, and (c) Both Absolute and Relative Error Control

As expected, these figures suggest that controlling fidelity via relative error limits is better suited to optimizing distortion metrics that quantify the error in each sample relative to the true sample value, while absolute error limits are better suited to optimizing distortion metrics that give the same weight to all sample reconstruction errors regardless of the true sample magnitude. Figure 4-29 gives a quantitative illustration of this effect by showing rate-distortion performance considering both Root Mean Square Error (RMSE) and Relative RMSE (RRMSE) distortion metrics when using absolute or relative error control. Here RMSE is defined as

$$\mathrm{RMSE} = \sqrt{\frac{1}{N_\mathrm{Z} \cdot N_\mathrm{Y} \cdot N_\mathrm{X}} \sum_{t,z} (s_z(t) - s'_z(t))^2}\,.$$

and RRMSE is defined as

$$\mathrm{RRMSE} = \sqrt{\frac{1}{N_\mathrm{Z} \cdot N_\mathrm{Y} \cdot N_\mathrm{X}} \sum_{t,z} \left(\frac{s_z(t) - s'_z(t)}{s_z(t)}\right)^2}\,.$$

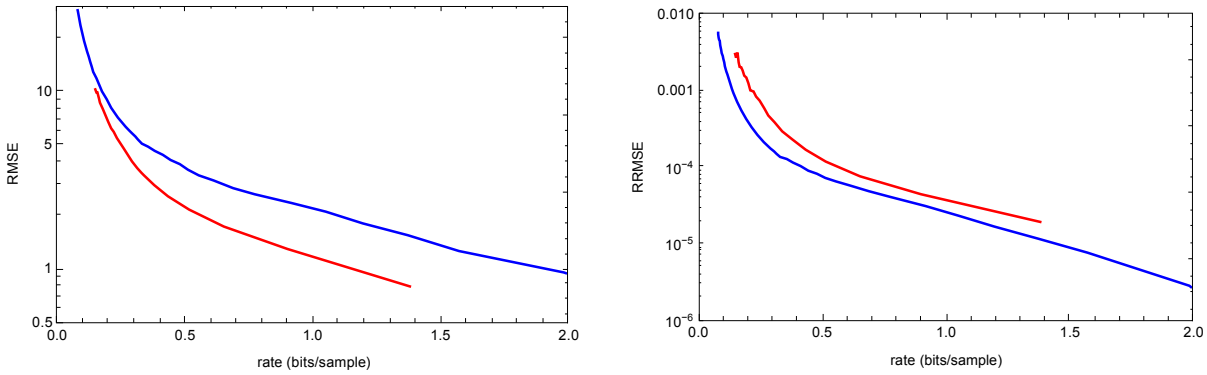


Figure 4-29: RMSE and RRMSE Distortion Using Absolute Error Control (Red) and Relative Error Control (Blue)

It should be noted that the use of relative error limits constrains the maximum error relative to the *predicted* sample magnitude $|\hat{s}_z(t)|$, rather than the true sample magnitude $|s_z(t)|$, because the true sample magnitude is not in general known to the decoder. This limitation can result in errors larger or smaller than expected when the predicted sample magnitude differs significantly from the true sample magnitude.

Providing the ability to control fidelity via the relative error limit is motivated by the fact that, in some applications, the impact of small reconstruction errors may depend on the signal level. For instance, even a low absolute error limit can lead to the removal of almost all the signal in darker image areas while still producing good results in terms of common global fidelity metrics such as PSNR, SNR, or MSE. This is a well-known issue in applications such as coastal imaging, where very dark water surfaces are present along with very bright land surfaces.

For many imaging sensors, noise is dominated by a combination of dark noise, a thermal phenomenon independent of the signal level, and photon or shot noise, which increases with the strength of the underlying signal. This innate noise can generally be treated as independent of the noise introduced by quantization during compression. Therefore, one may want to allow higher error limits for brighter areas to better match innate image noise while keeping lower error limits for darker areas to better preserve the information contained therein.

For a given pair of error limit values, using a combination of absolute and relative error limits can be an effective method of achieving high image quality over the full dynamic range, allowing a mission to meet an SNR requirement even in low luminance areas, at the price of higher compressed data rate compared to using either the absolute or relative error limit alone. This compromise is illustrated in figure 4-30, which was produced using the noise model from Pléiades images (see reference [27]) with total SNR, including both innate noise and compression-induced quantization noise, under different error limit control methods. By using both absolute ($A^* = 10$) and relative ($R^* = 200$) error limits, a minimum SNR of 25 dB can be achieved over the useful dynamic range of the data, including the low luminance range[4].

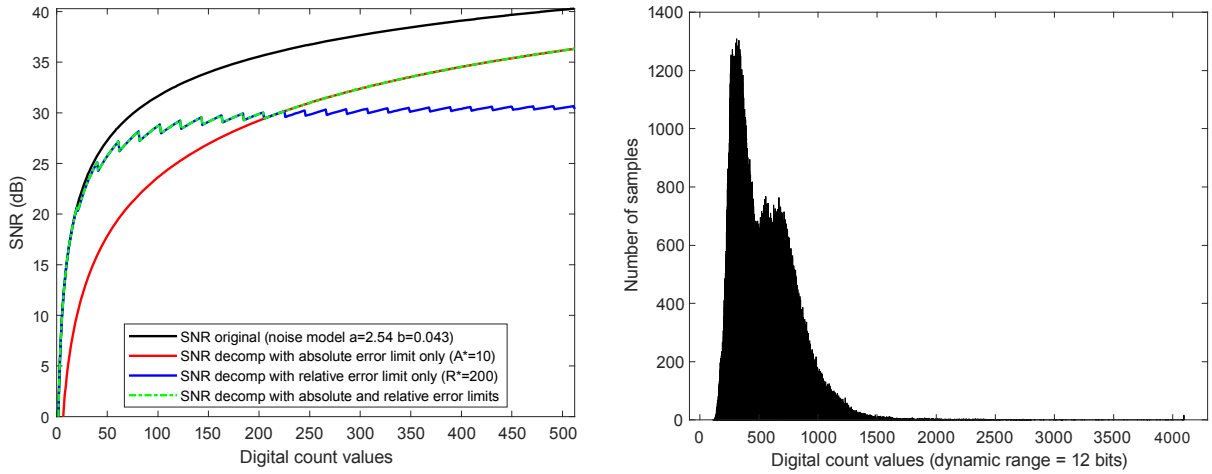


Figure 4-30: SNR of Original and Decompressed Images (Left) Using Absolute Error Limit $A^* = 10$ (Red), Relative Error Limit $R^* = 200$ (Blue) or Both (Green), for Data Simulating the Pléiades B2 Band, and Histogram of the Band (Right)

It should be noted that when the goal is to limit SNR loss over the full dynamic range, applying companding to the input image, especially with variance-stabilizing transforms such as the Generalized Anscombe Transform (see references [28] and [29]), followed by compression (lossless or near-lossless using absolute error limits alone), typically achieves better performance than that obtained by simply using relative and absolute error limits.

[4] Pléiades B2 images typically have useful dynamic ranges comprising mostly low luminance values, with average numeric counts near 500.

However, this generally requires precise knowledge of the noise model of the image sensor (which may differ for each spectral band) and additional onboard computational resources to preprocess the image data.

Figure 4-31 shows the benefit of the companding approach when applied to the second band of the Montpellier image. At an SNR loss target of 1.5 dB, the compressed data rate obtained from companding followed by lossless compression is 5.6 bits/sample, compared to 8.1 bits/sample when using both absolute and relative error limits (A^* = 10 and R^*= 25). At a 4 dB SNR loss target, the compressed bit rates are 4.6 and 8.0 bits/sample, respectively.

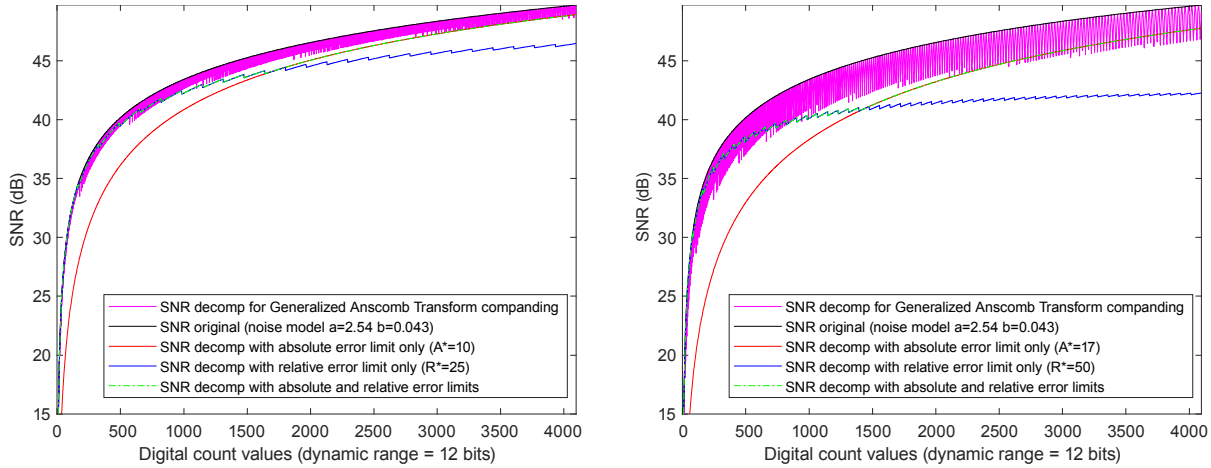


Figure 4-31: Comparison of SNR and Data Rates Achieved for Different Fidelity Control Approaches Applied to the Second Band of the Pléiades Montpellier Image with an SNR Loss Target of 1.5 dB (Left), and 4 dB (Right) over the Full Dynamic Range

4.2.7.2 Image Artifacts Arising from Large Error Limit Values

The quality of a reconstructed image band may be poor when compressed using an error limit value that is large compared to the signal energy in the band. As a severe example, figure 4-32 shows reconstruction of the first three bands of the 8-bit Landsat agriculture test image when compressed using $A^* = 30$, which yields a bit rate of 0.1 bits/sample.

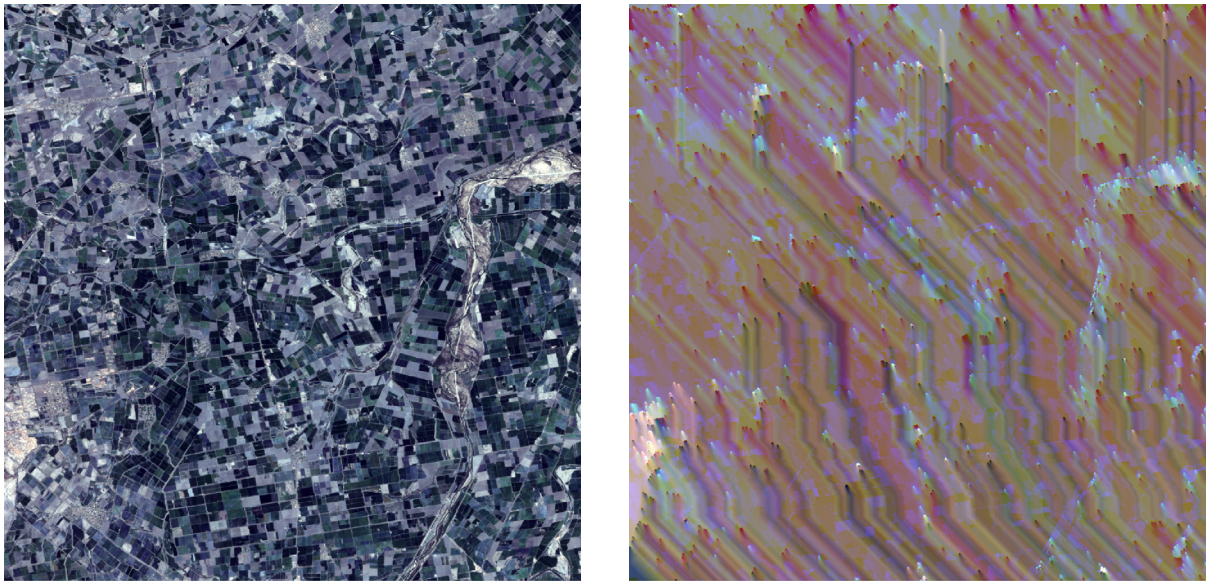


Figure 4-32: False-Color Image Derived from the First Three Bands of Landsat Image Original (Left) and Reconstructed Following Compression at $A^* = 30$ (Right)

This does not mean that dramatic artifacts will arise whenever one uses an absolute error limit as large as $A^* = 30$ or achieves a compressed data rate as low as 0.1 bits/sample. For example, figure 4-33 shows reconstruction of two different sets of bands in the AVIRIS Hawaii test image following compression using $A^* = 30$. All of the bands compress to 0.1 bits/sample. Bands (30, 33, 34) have standard deviation (21.2, 23.8, 26.5) and exhibit noticeable artifacts, while the trio of bands (78, 94, 17) have higher standard deviation (60.2, 57.9, 37.3) and do not exhibit artifacts.

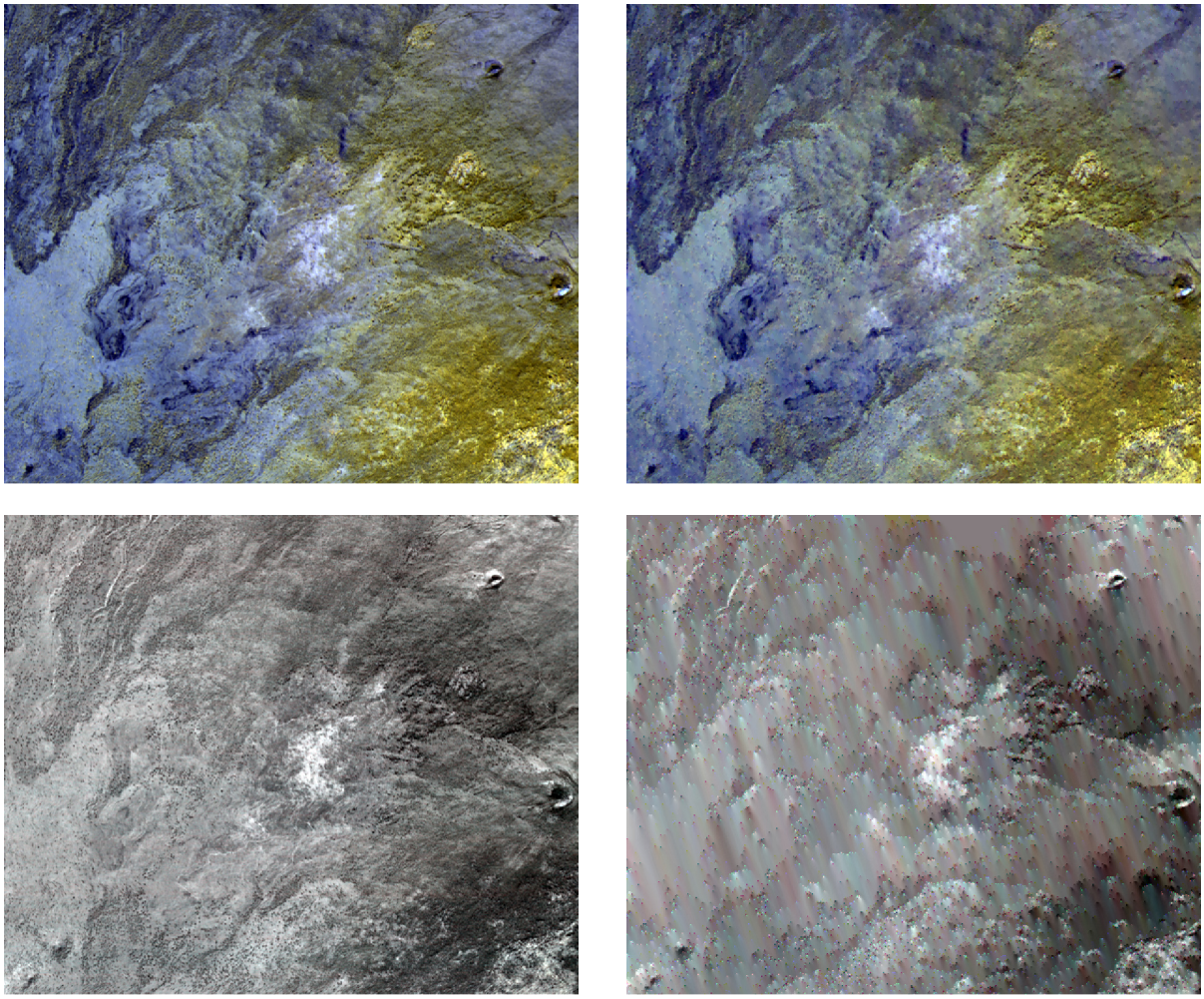


Figure 4-33: False-Color Images Derived from Bands 78, 94, 17 (Top Row) and 30, 33, 34 (Bottom Row) of AVIRIS Hawaii Original Image (Left Column) and Reconstructed Following Compression at $A^* = 30$ (Right Column)

4.2.7.3 Periodic Error Limit Updating and Fixed-Rate Control

Predictive image coding methods for hyperspectral images are particularly attractive because of the high performance achieved at a low computational complexity. However, they inherently produce variable compressed data rates depending on the content of the images, which can be a drawback from a system-level standpoint in which deterministic fixed-rates are usually preferred for easier mission management (data storage and delivery).

To overcome such limitations, it is possible to use rate-control mechanisms to produce bitstreams that respect an overall bit budget constraint. The main obstacle for rate control methods in prediction-based coding systems is the complex relationship between the quantized prediction residuals and compressed data rate. The Recommended Standard does not define any particular strategy to achieve such fixed-rate control, but contains provisions that may allow the use of such techniques. Accordingly, several rate-control methods

modelling bit rate versus quantizer settings have already been proposed by making use of those provisions (see references [30] and [31]).

This rate-control capability is the rationale for including periodic error limit updating as a feature of the Recommended Standard. When used, the user may change the error limit values at some user-specified frame interval, thus allowing quantizer parameters to adapt as needed to meet a desired target data rate.

It should be noted that making large changes in error limit values within an image can be expected to produce large variations in reconstructed image quality. In particular, as shown in 4.2.7.2, compression artifacts may become conspicuous when quantizer step size is large compared to signal energy.

4.3 ENTROPY CODER

4.3.1 INTRODUCTION

At high rates the hybrid encoder encodes most samples using a family of codes that are equivalent to those used by the sample-adaptive coder, and thus has nearly identical performance. But the hybrid encoder has substantially better performance than the sample-adaptive and block-adaptive entropy coders at low bit rates (figure 4-34). The sample-adaptive encoder is unable to reach rates lower than 1 bit per sample, whereas the block-adaptive encoder is able to do so but with non-negligible rate overhead. In addition, the block-adaptive encoder generally has poor performance when encoding in Band-Interleaved-by-Pixel (BIP) order, because samples from different bands (which may have different statistical behavior) are jointly encoded in the same block. Using Band-Sequential (BSQ) or Band-Interleaved-by-Line (BIL) encoding order somewhat mitigates this issue.

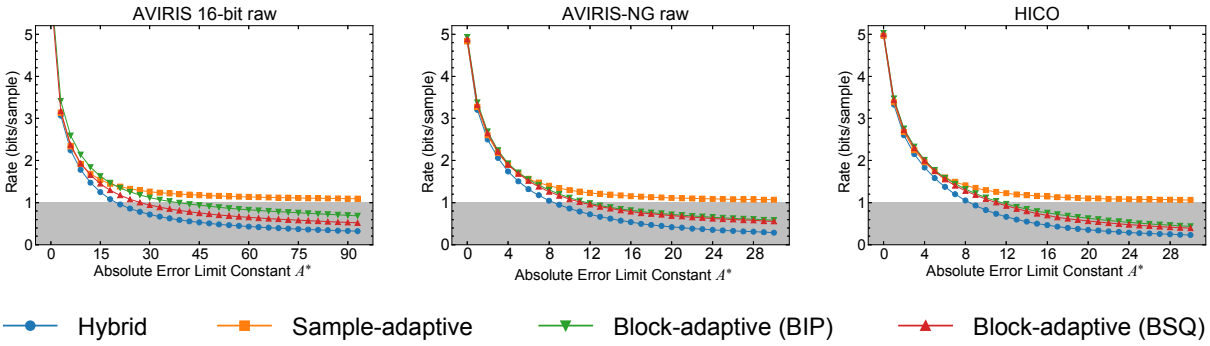


Figure 4-34: Compressed Data Rate for Each Entropy Coder as a Function of Absolute Error Limit Constant A^*

Subsections 4.3.2, 4.3.3, and 4.3.4 discuss some of the key parameters associated with the sample-adaptive, hybrid, and block-adaptive encoding approaches, respectively. Differences in implementation between the entropy coding options are discussed in 5.8.3.6.

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; Σ_z is initialized in a way that ensures that the GPO2 code parameter for the band is initially equal to the user-specified parameter k'_z. Thus k'_z could be assigned using a training image (e.g., one could set k'_z equal to the last GPO2 code parameter used in band z from the most recently compressed image). An accumulator initialization constant K may be used, in which case $k'_z = K$ 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.

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, especially when γ^* is small. To illustrate this, figure 4-35 shows, for two sample images, the range of possible bit rates achievable by varying the values of (γ_0, γ^*, K) from best to worst after partitioning each image along the y axis into smaller images. One can observe from the figure that optimizing the sample-adaptive coding parameters provides much less benefit when image spatial size is large. For this reason, in the sequence considered, images are partitioned along the y axis into smaller images, each with height $N_Y = 16$, except possibly the last image.

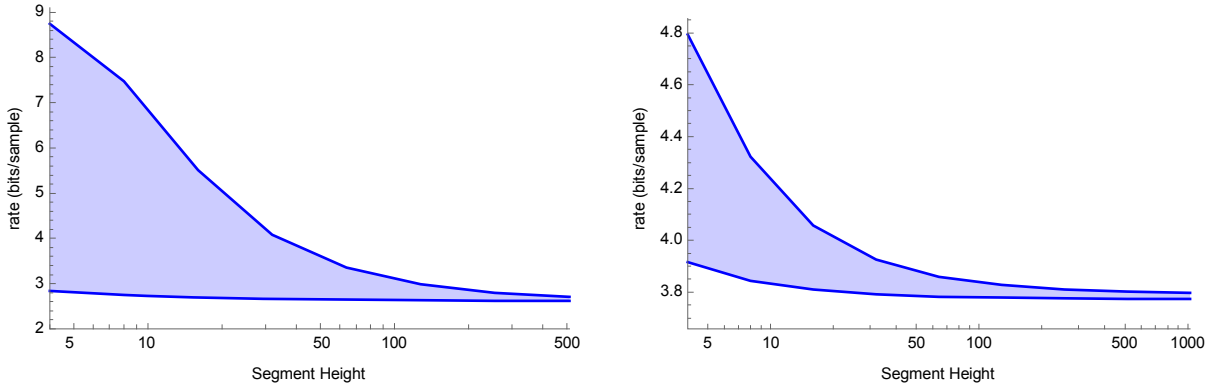
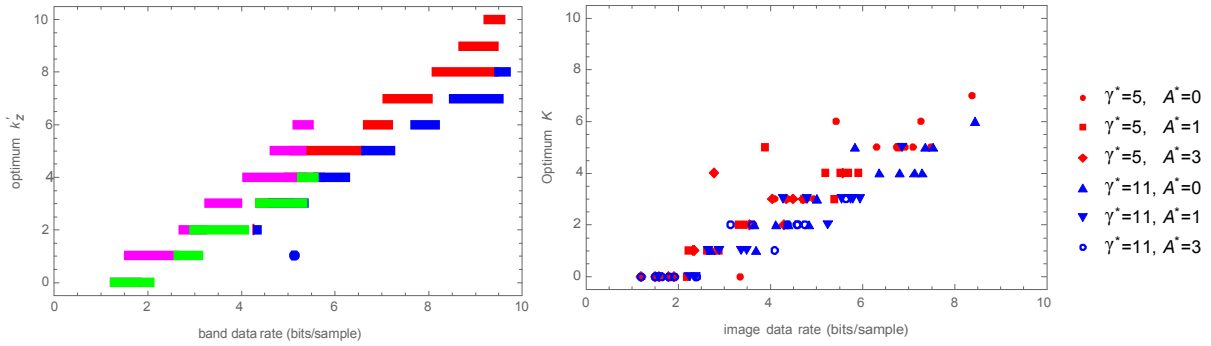


Figure 4-35: Range of Possible Lossless Compressed Data Rates Achieved by Varying (γ_0, γ^*, K), as a Function of Image Height, for the AVIRIS Hawaii Image (Left) and Landsat Mountain Image (Right)

Each GPO2 code is intended to work well for some limited range of source entropy. When the optimum value of k'_z is used to compress each band of an image, bands with larger optimum k'_z value tend to be compressed to higher bit rates. A smaller effect is that the

optimum value of k_z' tends to be larger when γ^* is smaller. Figure 4-36 (left) illustrates these effects for an AVIRIS-NG image, and figure 4-36 (right) illustrates a similar result relating the optimum value of accumulator initialization constant K to overall bit rate. The value of γ_0 appears to have little influence on the optimum value of k_z'.



Left: Range of bit rates covering 90 percent of the bands for which a given k_z' value is optimum for a calibrated AVIRIS-NG image partitioned into segments with height N_Y = 16 frames and compressed using (γ^* = 5, A^* = 0) (red), (γ^* = 11, A = 0) (blue), (γ^* = 5, A = 8) (magenta), and (γ^* = 11, A = 8) (green).

Right: optimum accumulator initialization constant K versus data rate for 15 different images partitioned into 16-frame segments, at different combinations of γ^* and A. In each case γ_0'=4.

Figure 4-36: Effects When Optimum Value of k_z' Is Used to Compress Each Band of an Image

Figure 4-37 shows rate as a function of k_z' for two bands of an image when either γ_0 or γ^* is fixed. Observe that performance is more sensitive to k_z' when γ^* and/or γ_0 is large. Performance is generally improved by avoiding the largest allowed values of k_z'. Using a smaller value of k_z' not only tends to improve compression effectiveness, but also seems to generally make compression performance less sensitive to the choice of γ_0 and γ^*. The use of a smaller value of γ^* causes the impact of the values of γ_0 and k_z' to diminish more rapidly and offer improved performance, particularly when a poor choice of k_z' is used. Figure 4-38 illustrates similar results for overall compressed data rate and K.

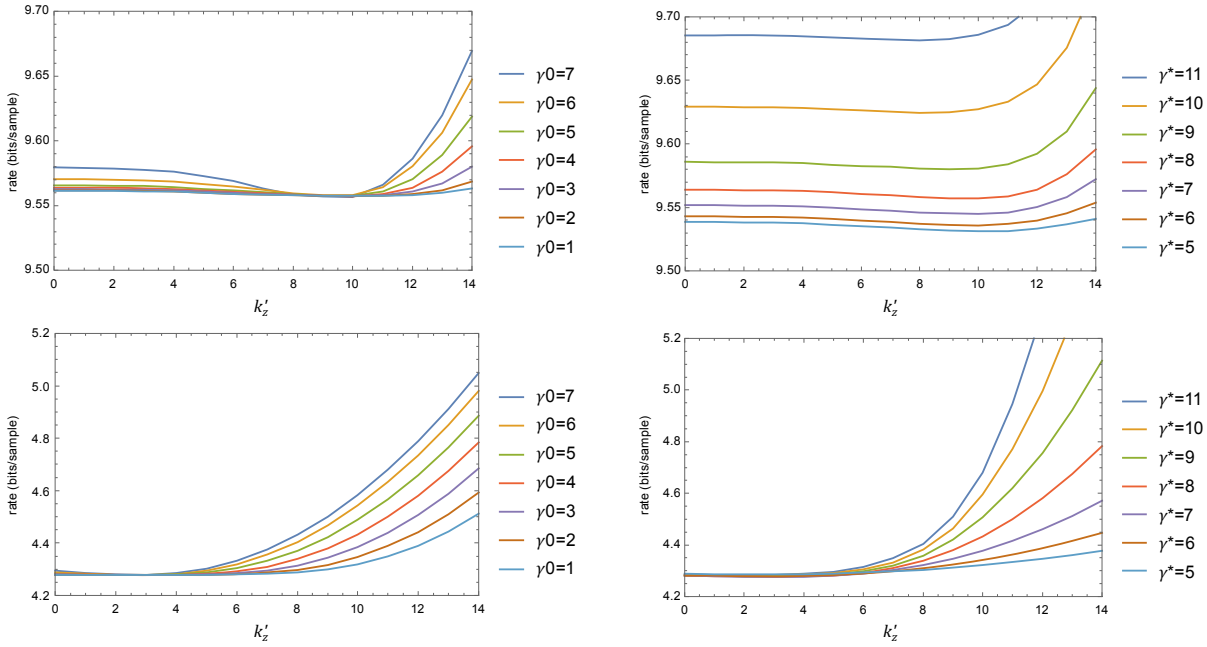


Figure 4-37: Data Rate As a Function of k'_z for Band 61 (Top) and 308 (Bottom) for Lossless Compression of the Calibrated AVIRIS-NG Image with $\gamma^* = 8$ (Left) and $\gamma_0 = 4$ (Right)

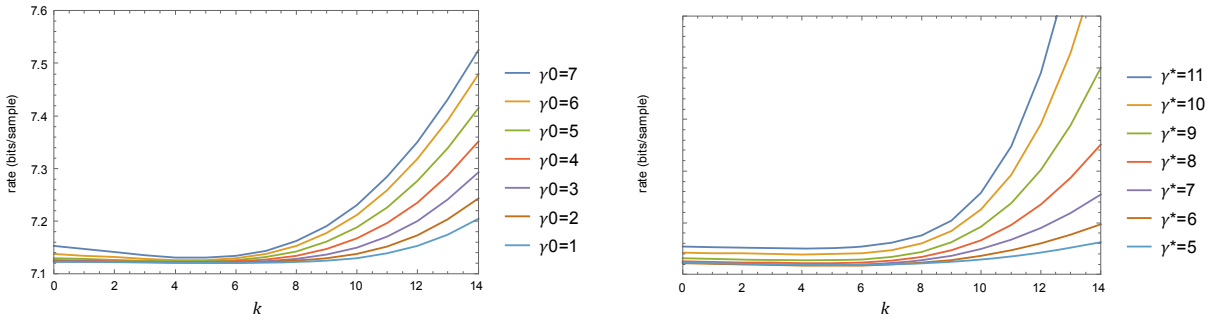


Figure 4-38: Data Rate As a Function of *K* for Lossless Compression of the Calibrated AVIRIS-NG Image with $\gamma^* = 8$ (Left) and $\gamma_0 = 4$ (Right)

Figure 4-39 compares different combinations of entropy coding parameters when used to compress an image comprised of the second 16-frames from several different test images, in each case applying one of four different parameter selection strategies: (a) using the smallest allowed parameter values ($\gamma^* = 4$, $\gamma_0 = 1$, $K = 0$), (b) setting $K = 0$ and using the optimum combination of (γ^*, γ_0), (c) setting $\gamma^* = 4$ and $\gamma_0 = 1$ and selecting each k'_z from the estimated mean mapped quantizer index at the end the preceding 16-frame image segment, and (d) selecting each k'_z from the estimated mean mapped quantizer index at the end the preceding 16-frame image segment with the optimum combination of (γ^*, γ_0). In each case, the performance is shown relative to the bit rate obtained by simultaneously optimizing (γ^*, γ_0) and each k'_z. It should be noted that even the simplest of these strategies provides an extremely small reduction in performance compared to optimum, less than 1% for all of the cases tested.

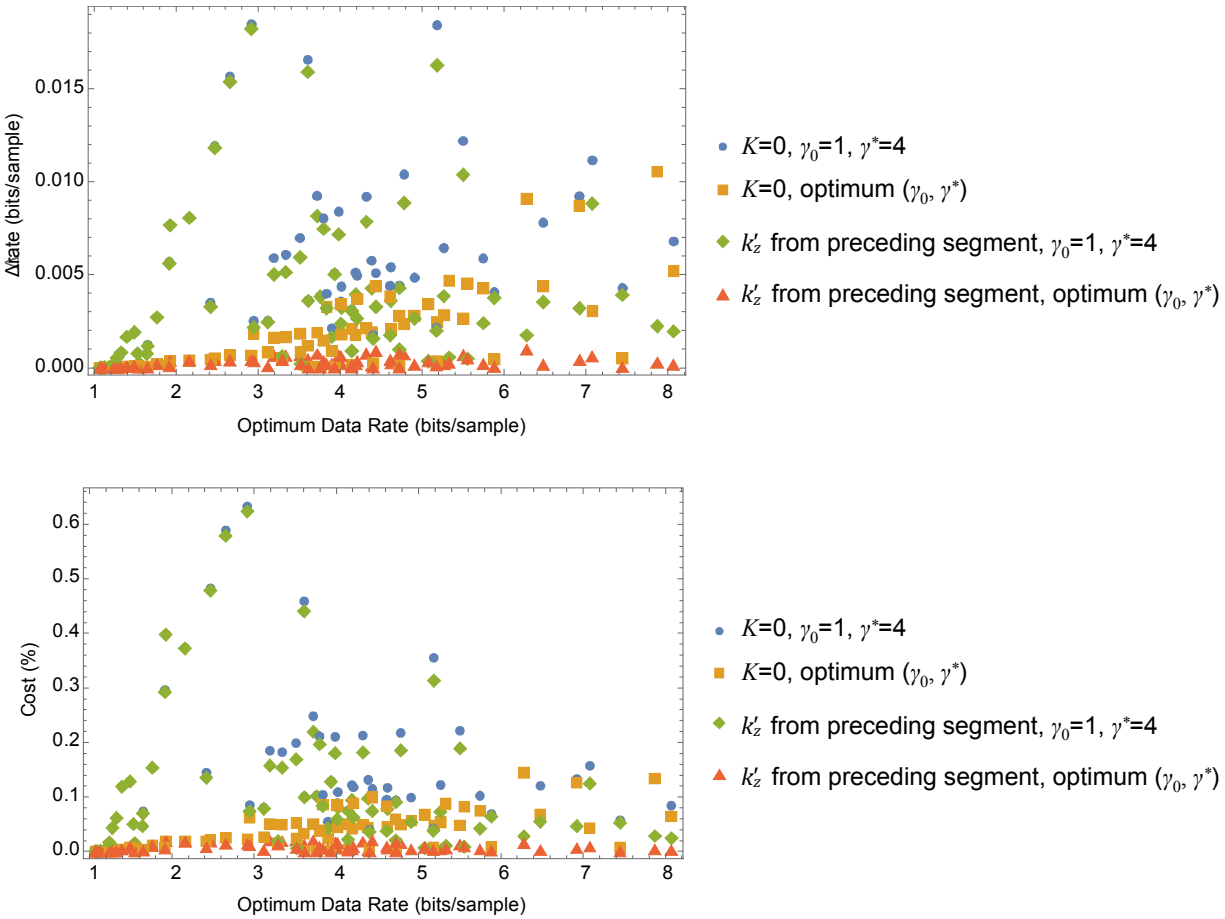


Figure 4-39: Comparison of Different Strategies for Selecting Sample-Adaptive Coding Parameters for the Second 16-Frame Segment of AVIRIS-NG A rad1, HICO New York, AVIRIS Hawaii, Landsat Mountain, MODIS 500m, Pléiades Montpellier, and Perpignan Images, Compressed at A^* Values 0 through 8

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 quantizer index 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 index of the code used to encode a sample, that is, the value of k, is selected based on an estimate of the mean value of the mapped quantizer index in the current spectral band, as described in 3.3.4. Smaller mapped quantizer index 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 4-40 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 $U_{max} = 20$.

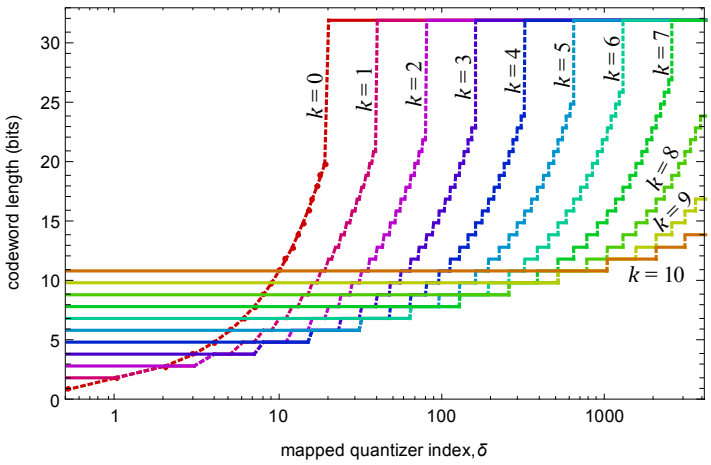


Figure 4-40: Codeword Lengths for the Set of Length-Limited GPO2 Codes Used by the Sample-Adaptive Entropy Coder When Image Dynamic Range Is $D = 12$ Bits and the Unary Length Limit Is $U_{max} = 20$

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 quantizer index), in return for less efficient coding of some smaller values of δ, as can be observed in figure 4-41, 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.

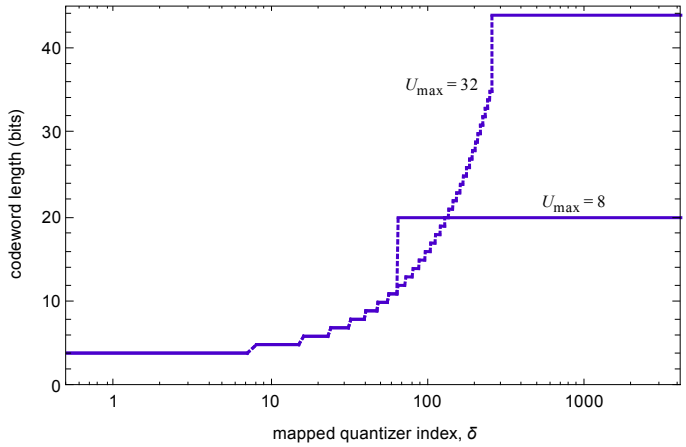


Figure 4-41: Codeword Lengths for k=3 Length-Limited GPO2 Codes Used by the Sample-Adaptive Entropy Coder at Minimum and Maximum Allowed Values of the Unary Length Limit U_{max} When Image Dynamic Range Is $D = 12$ Bits

Because the sample-adaptive encoder adaptively selects the code index 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 4-42 shows that, for lossless compression, 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).

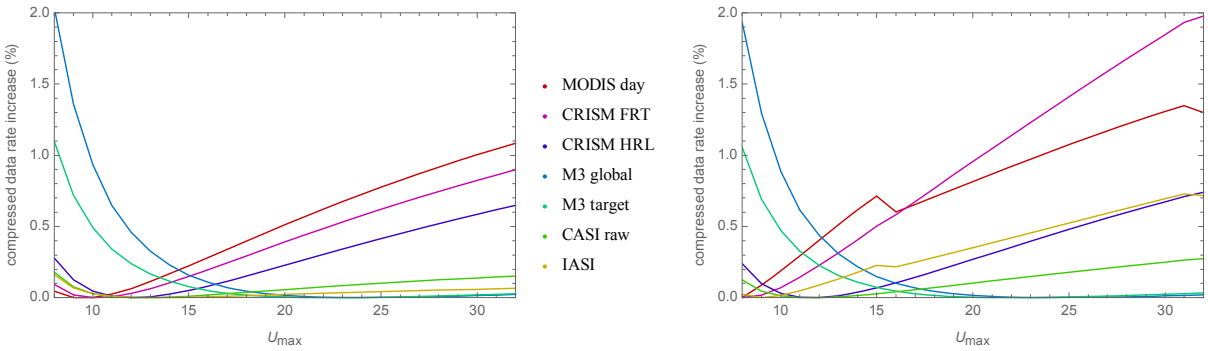


Figure 4-42: Impact of Changing the Unary Length Limit on Compressed Data Rate Shown for Maximum and Minimum Allowed Register Size R (Left and Right, Respectively) for Lossless Compression Using Backwards-Compatible Settings

4.3.3 HYBRID ENTROPY CODER

Code selection statistics used by the hybrid entropy coder are periodically rescaled at an interval determined by parameter γ^*. This parameter has the most significant impact on hybrid entropy coder performance. Within each band, each time rescaling occurs, an additional bit is output to the compressed bitstream so that the decompressor can reconstruct the code selection statistics. At lower bit rates, this overhead can become significant, and larger values of γ^* are needed to diminish its impact. This can be seen in figure 4-43 where γ^* is varied when the absolute error limit constant A^* is adjusted (via trial-and-error) to achieve bit rates of approximately 0.1, 0.5, 1, 2, 3, and 4 bits per sample.

The results suggest the use of $\gamma^* \approx 6$ for high rate, and to transition to larger γ^* values at rates below 1 bit per sample.

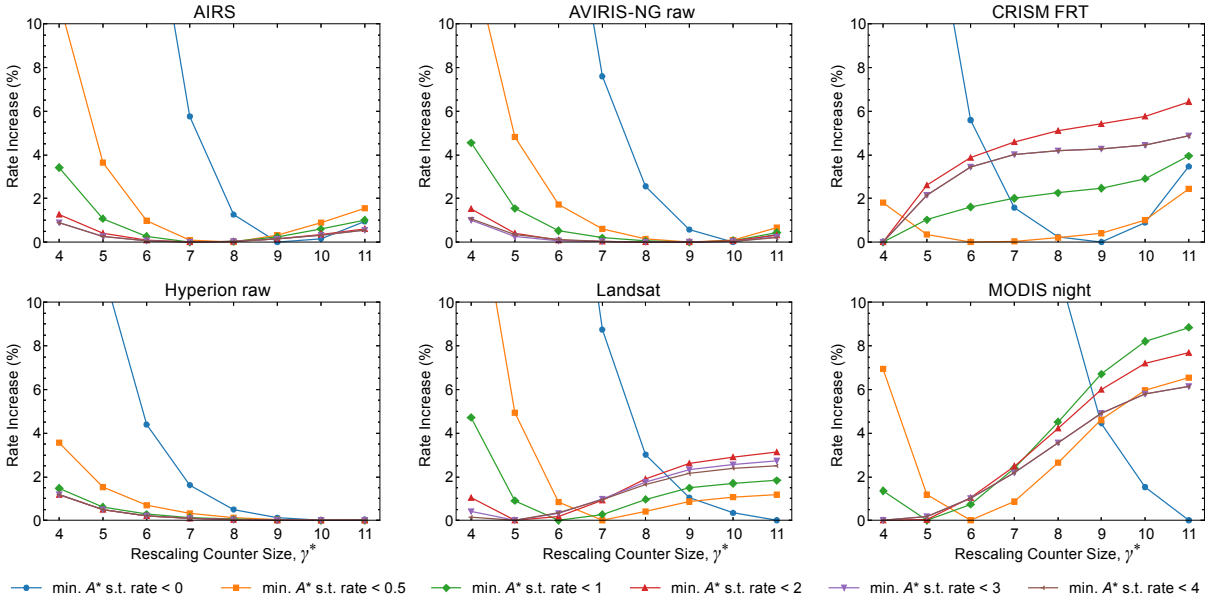


Figure 4-43: Change in Compressed Data Rate, Relative to Optimal Choice of γ^*, as a Function of Rescaling Counter Size γ^*

4.3.4 BLOCK-ADAPTIVE ENTROPY CODER SETTINGS

4.3.4.1 Overview

The block-adaptive encoding method partitions the mapped quantizer indices 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.

4.3.4.2 Sample Encoding Order

Under the block-adaptive encoding approach, the set of mapped quantizer indices 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 as can be seen in figure 4-34, compressed data rate depends on the sample encoding order when the block-adaptive encoder is used; this is not the case when the sample-adaptive or hybrid encoder is used.

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

- BIP encoding order generally does not perform as well as BIL or BSQ. (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 quantizer indices is all zeros (which arises when the prediction errors for all samples in the block are smaller than the quantizer bin size), the block-adaptive encoder's 'zero block' coding method is used (see subsection 3.4.3 of reference [14]). 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.

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 differ because all-zero blocks are more likely to be adjacent under BSQ order than BIL order.

4.3.4.3 Block Size

Under the block-adaptive coding approach, mapped quantizer indices 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, when $D \geq 9$ 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.

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 4-44 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 lossless compression, presumably because it increases the fraction of blocks that are all zeros and are thus economically encoded using the zero-block coding method.

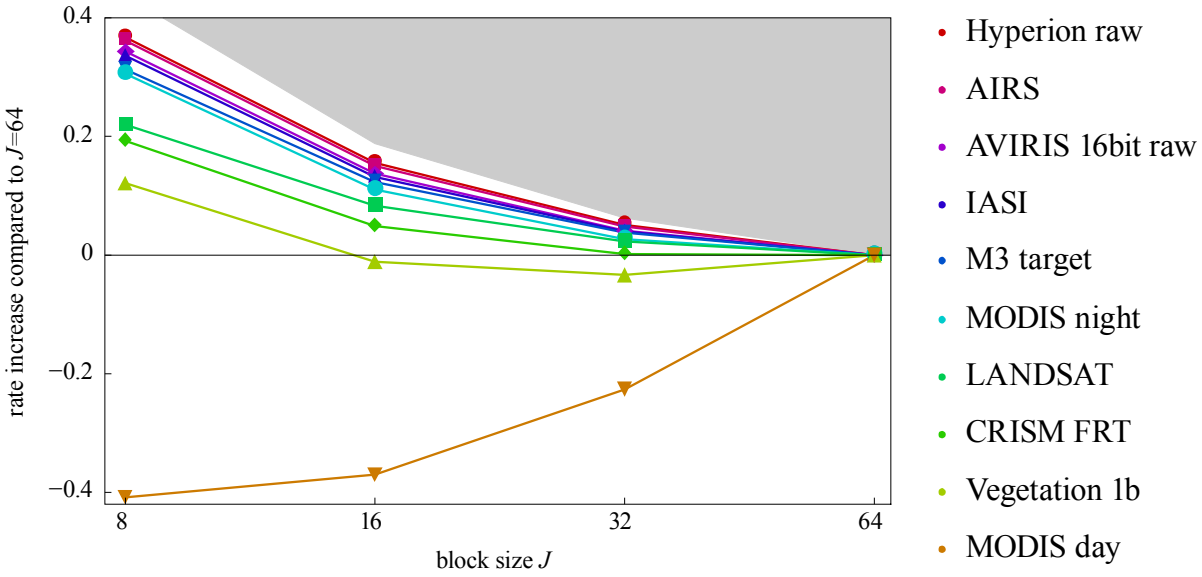


Figure 4-44: Average Increase in Compressed Bit Rate, Compared to the Use of Block Size $J = 64$, under Block-Adaptive Coding Using BSQ Sample Order for Lossless Compression Using Backwards-Compatible Settings

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.

Regarding the block size values, figure 4-45 indicates that for small values of the absolute error limit constant, where the performance penalty of using the block-adaptive encoder instead of the hybrid encoder tends to be smaller, employing a block size of 64 yields the best performance, though this trend is reversed for larger A^* values.

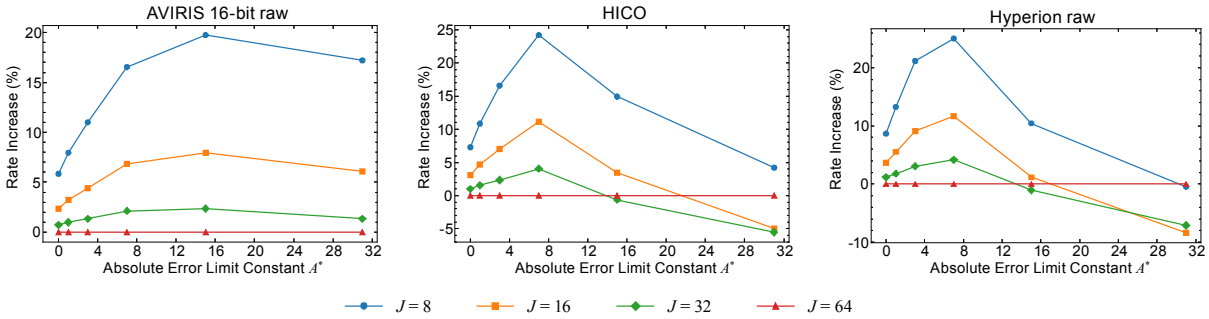


Figure 4-45: Change in Compressed Data Rate Using the Block-Adaptive Entropy Coder with Different Block Sizes, J, as a Function of Absolute Error Limit Constant A^*

NOTE – BIL encoding order is used. Performance is shown relative to J = 64.

4.3.4.4 Reference Sample Interval

The *reference sample interval* parameter, *r*, affects how 'zero-block' runs (sequences of blocks of zero values) are encoded. However, as can be seen from figure 4-46, the practical effect of this parameter is negligible except for combinations of very small values of *r* and large values of *A**.

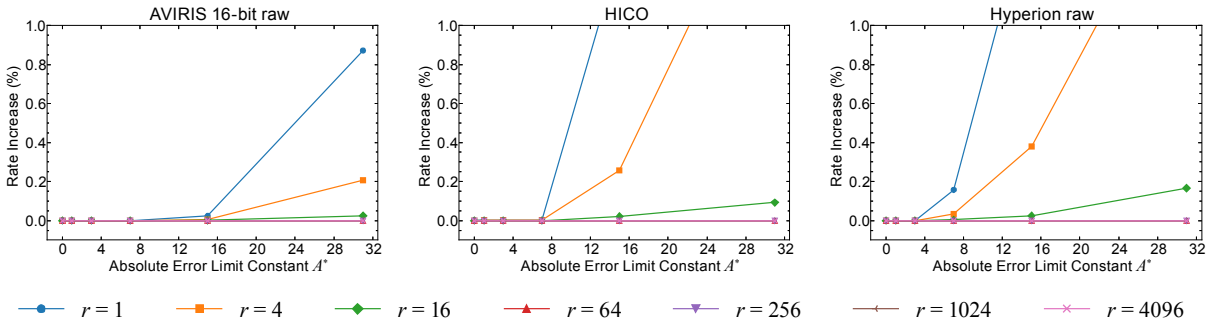


Figure 4-46: Average Compressed Data Rate Using the Block-Adaptive Entropy Coder As a Function of Absolute Error Limit Constant A^*

NOTE – Performances is shown relative to $r = 4096$. BIL encoding order is used.

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 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. Subsection 5.5 shows the potential improvement in compression effectiveness produced by reordering spectral bands in an image. Subsection 5.6 examines the impact of spatial misregistration on compression performance. Subsection 5.7 shows that, for some distortion metrics, rate-distortion performance may be improved by selecting a sample reconstruction point that is not at the center of the quantizer bin. Subsection 5.8 discusses hardware implementation considerations.

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 the loss of most or all of the image.

When the hybrid entropy coder is used, as described in 3.3.5, the decompressor reconstructs mapped quantizer indices in reverse order. Once the mapped quantizer indices have been decoded, reconstruction of image samples proceeds in forward order. Thus a single bit error in the middle of a compressed image would nearly always prevent accurate reconstruction of mapped quantizer indices at the beginning of the image, thus preventing reconstruction of the entire image.

When the sample-adaptive or block-adaptive coders are used, 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 losslessly-compressed hyperspectral image following a single bit error occurring near the midpoint of the compressed image data.

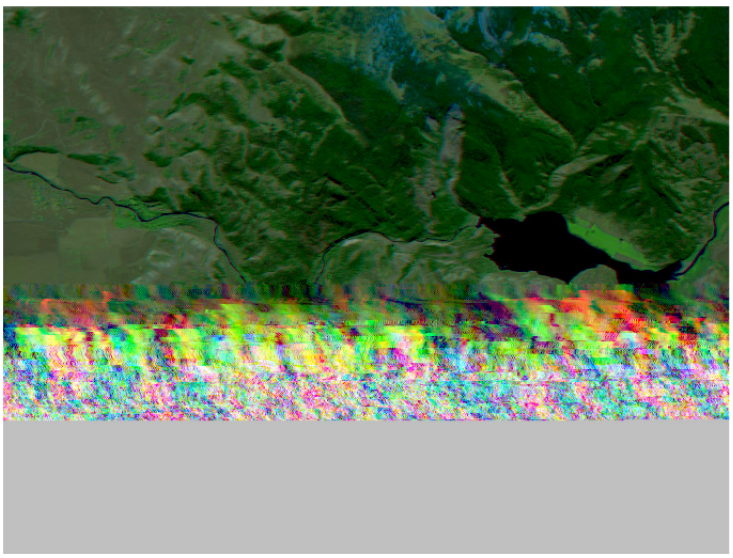


Figure 5-1: Example of the Impact of a Bit Error in the Compressed Image

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.

If the sample-adaptive or block-adaptive coders are used, 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 [2]), *encapsulation packets* (reference [32]), *files* (reference [3]), or *transfer frames* (references [4] and [5]). This may lead to the selection of appropriate channel codes (references[33], [7], [34], and [35]) 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) 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.

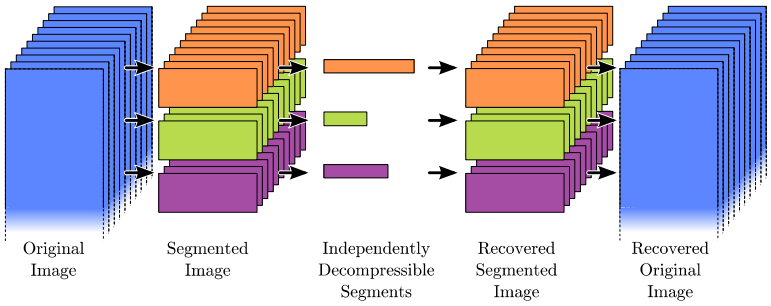


Figure 5-2: Overview of Image Segmentation

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).

To illustrate that compression effectiveness tends to increase by using larger segments and by using custom weight initialization, lossless 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.

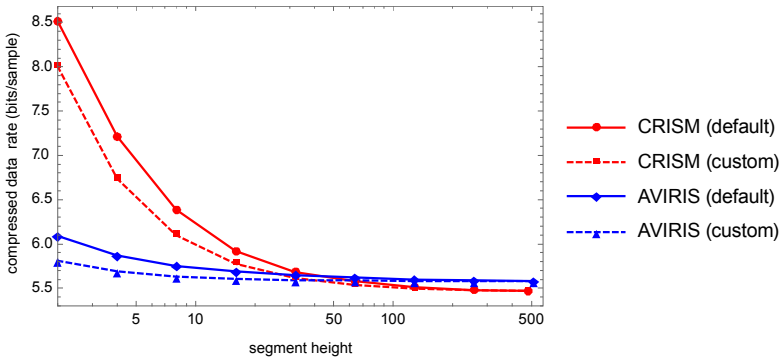


Figure 5-3: Lossless Compressed Data Rate as a Function of Segment Height for Test Images 'crism_frt00013e49_07_sc166' and 'aviris_sc10_raw' Using Backwards-Compatible Compression Settings

In addition, an implementer could segment the hyperspectral image along the *x*- and/or *z*-axes. Partitioning along these directions may provide an implementation advantage. Specifically, each segment could be processed separately in parallel (to increase throughput) or sequentially with a smaller compressor (to reduce hardware resource usage). Of course, for the reasons noted above, such partitioning generally decreases compression effectiveness.

To compare the effects of partitioning along different axes, three hyperspectral images are partitioned into segments along one of the axes. Figure 5-4 shows the compressed data rate achieved as we vary segment size when partitioning in the *x*, *y*, and *z* directions.

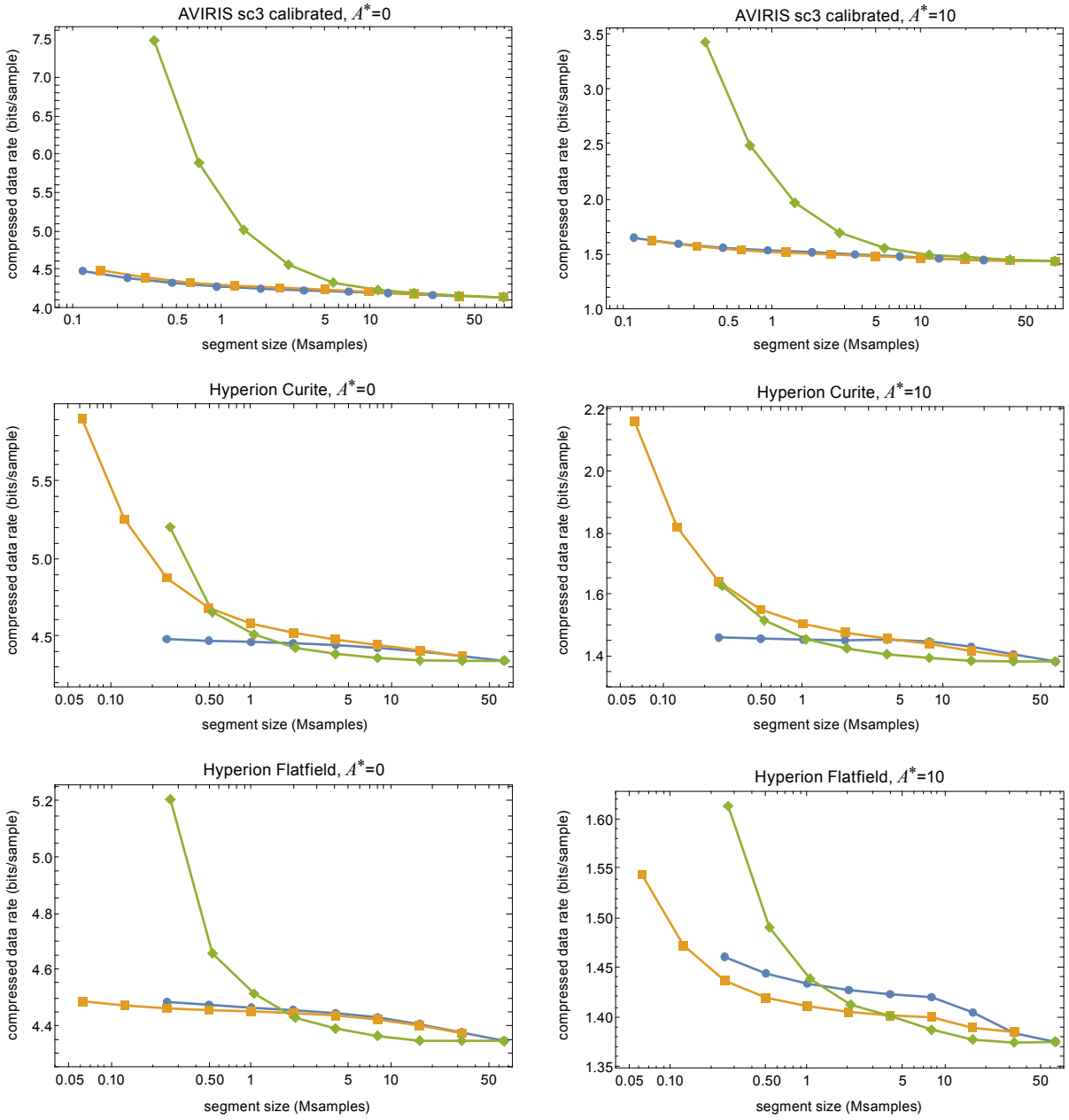


Figure 5-4: Compressed Data Rate versus Segment Size When Partitioning along *x*, *y*, and *z* Axes (Blue, Yellow, and Green, Respectively) at Different Absolute Error Limit Values

For the images examined, compression performance as a function of segment size is nearly identical for *x* and *y* axes partitioning. Performance degrades much more quickly as we decrease segment size when partitioning along the *z* axis. For the raw Hyperion image, the *x* and *y* axes do not exhibit the same symmetry. Partitioning along the *x* axis provides better performance because the image includes streaking artifacts parallel to the *y* axis.

The results suggest that the increase in compressed data volume arising from partitioning an image into segments may be modest if the partitioning direction is well chosen and the image segments are not very small.

It should be noted that the boundaries between image segments may become visible in the reconstructed image when compression is not lossless and the compressed data rate is very low. As an example, figure 5-5 illustrates an artifact evident at the boundary between two segments of an image compressed to 0.14 bits/sample.

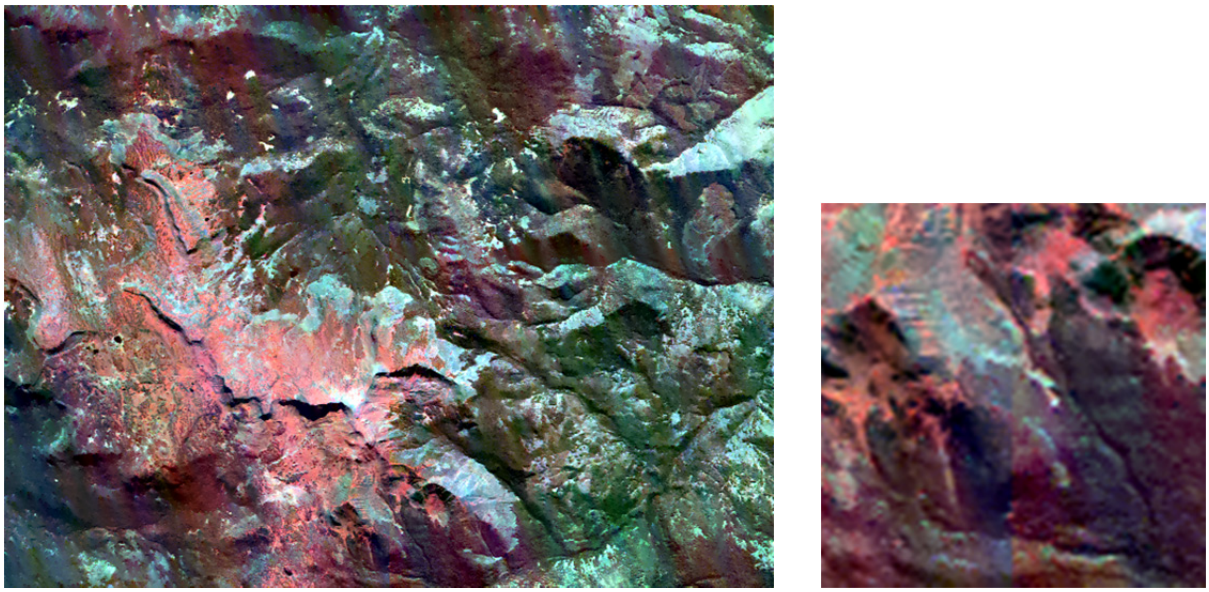


Figure 5-5: False-Color Image Derived from Bands 82, 94, and 105 of a Calibrated AVIRIS Image Reconstructed Following Compression at $A^* = 130$

NOTE – In the detail image (right), the boundary between left and right halves appears as a color shift.

5.4 DETECTOR NONUNIFORMITY CORRECTION

As discussed in subsection 4.2.1.3 (see figure 4-6), 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 more effective compression can be expected from the elimination or reduction of such artifacts prior to compression.

Performing complete radiometric calibration on board 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 when compression is lossless. An illustrative example follows.

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

$$s'_{z,y,x} = s_{z,y,x} - b_{z,x}\,. \tag{24}$$

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

$$b_{z,x} = \mathrm{Round}\left[\, m_{z,x} - \mathrm{median}\left[\, \{m_{z,x}\}_{x=0}^{N_\mathrm{X}-1} \,\right]\right], \tag{25}$$

where

$$m_{z,x} = \frac{1}{N_\mathrm{Y}} \sum_{y=0}^{N_\mathrm{Y}-1} s_{z,y,x}\,. \tag{26}$$

Figure 5-6 and figure 5-7 show false-color images derived from raw and offset-adjusted versions of CRISM and M3 images using the method of equation (25). Figure 5-8 shows a similar result for a Hyperion image, using an offset array provided by the mission.[5] 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 5-7 (caused by the prominent dark feature in these columns of the image combined with the somewhat simplistic calculation of equation (25)) should not cause undue concern because the offset adjustment is reversible after decompression.

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

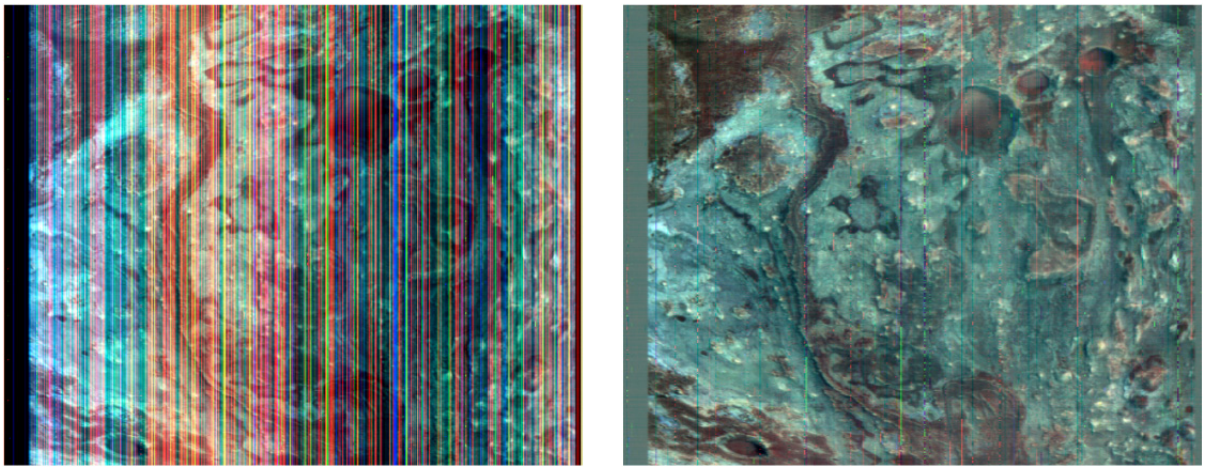


Figure 5-6: False Color Images Derived from Bands 200, 300, and 400 of Raw (Left) and Offset-Adjusted (Right) Versions of CRISM FRT Image 00009326_07_sc167

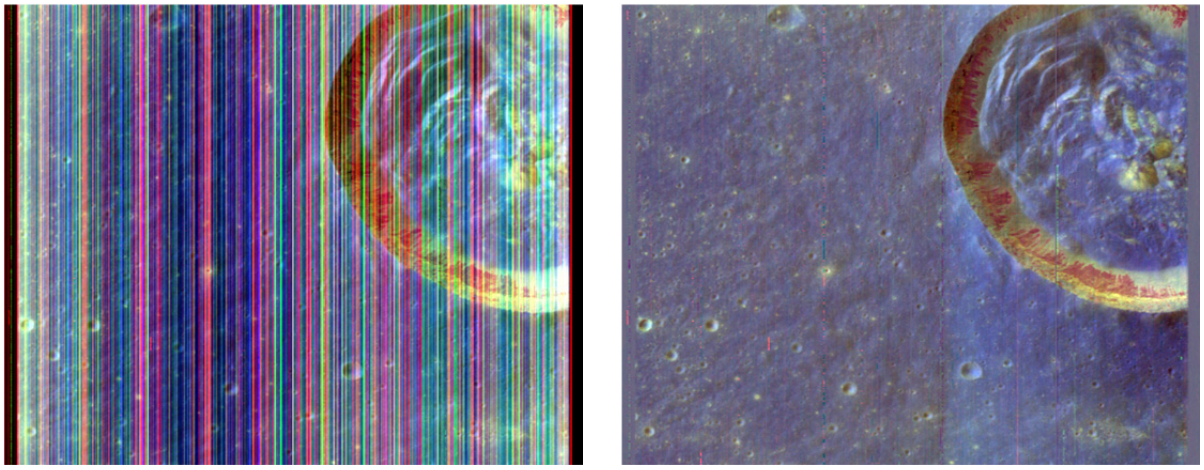


Figure 5-7: False Color Images Derived from Bands 50, 150, and 200 of Raw (Left) and Offset-Adjusted (Right) Versions of M3 Target Image A

The offset adjustment dramatically decreases streaking artifacts in these three examples. But some remaining streaking artifacts are evident in the images of figure 5-6 and figure 5-7, including some that are clearly not well modeled as a constant additive term.

Figure 5-9 shows the compressed data rates for the raw and offset-corrected images shown in figure 5-6, figure 5-7, and figure 5-8 for all four combinations of prediction mode and local sum type using backwards-compatible lossless compression settings.

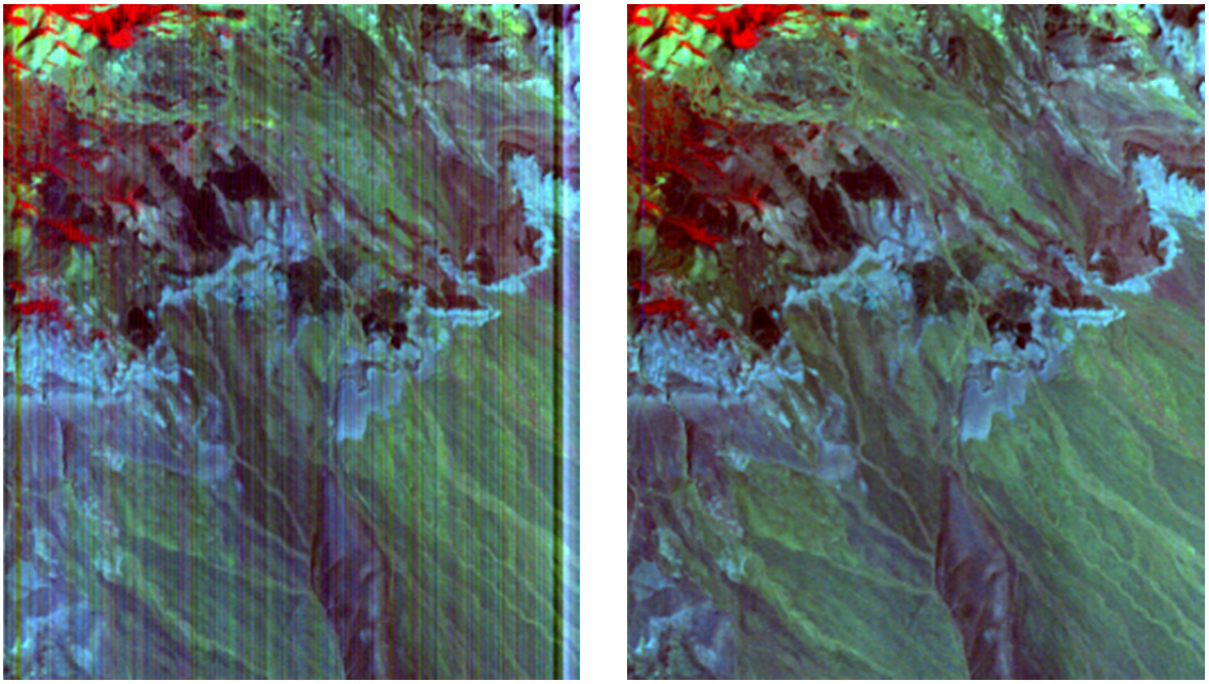


Figure 5-8: False Color Images Derived from Bands 83, 138, and 200 of Raw (Left), and Offset-Adjusted (Right) Versions of Hyperion Cuprite Image

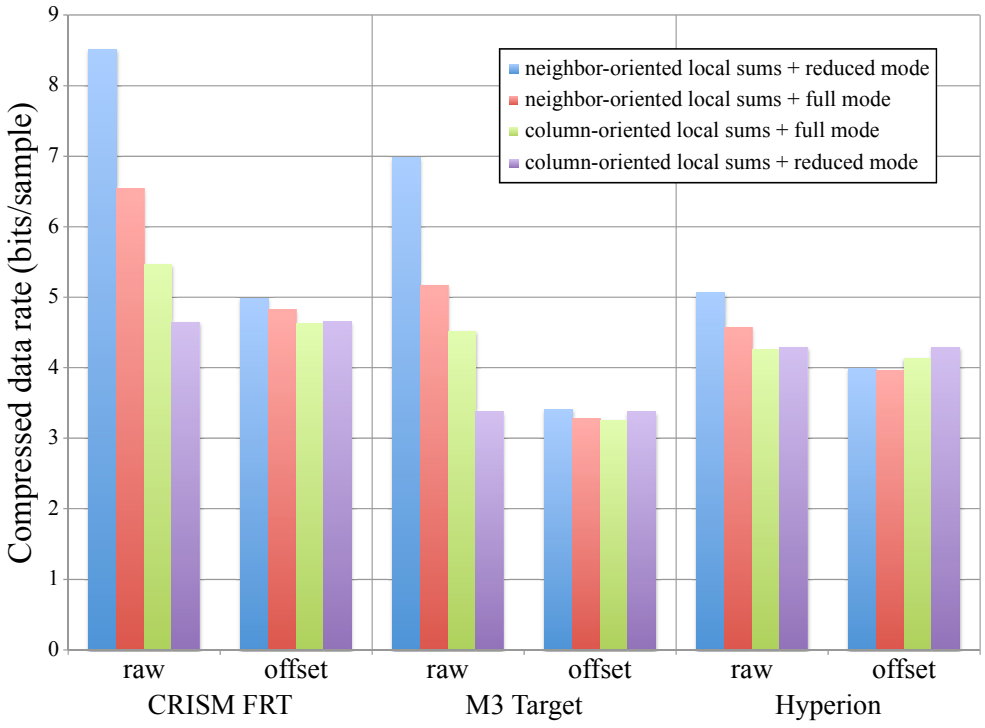


Figure 5-9: Compressed Data Rates in Bits/Sample for Raw and Offset-Adjusted Images CRISM FRT 00009326_07_sc167, M3 Target A, and Hyperion Cuprite

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 5-9 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 5-6.

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.

5.5 BAND RE-ORDERING

Most multi- and hyperspectral image compression algorithms follow the ‘natural’ band ordering; that is, the different bands are compressed in order of increasing wavelength. The idea of band re-ordering (reference [36]) is that this natural ordering may not optimize 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 that minimizes bit rate, which is intimately related to the compression method and can be a complex problem given the large number of possible band orderings. In most cases, the optimal adaptive on-line computation may not be feasible on board. Instead of optimizing band ordering for each image, one could search for a band ordering that works well for a given image sensor. A fixed reordering computed off-line can provide near-optimal performance (reference [36]). 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 [37].

As far as this 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. 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. A one-dimensional supplementary information table (subsection 3.5 of reference [1]) could be used to indicate band order.

– In general, the use of band reordering has been shown to provide benefits for this Recommended Standard (see reference [37]) as well as for other lossless multispectral and hyperspectral image compressors (reference [38]). 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 [37] 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 [36] and [37]) 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.

5.6 IMPACT OF MISREGISTRATION

This subsection provides examples illustrating the impact of spatial misregistration on lossless compression performance (compressed data rate in bits/sample). Here, misregistration refers to spatial misalignment from one band to the next.

Two different methods are used to simulate different amounts of misregistration:

sub-pixel misregistration, that is, misregistration less than 1 pixel, is simulated using higher resolution images, and misregistration by more than one pixel is simulated by translating bands of the original images.

5.6.1 SUB-PIXEL MISREGISTRATION

Data used to illustrate the impact on compression performance of sub-pixel misregistration are produced from high resolution simulated Pléiades images (see annex A). Pléiades multispectral images are composed of 4 spectral bands, and the misregistered image is produced by translating each band by (x,y) offsets denoted B_0, B_1, B_2, and B_3, respectively. Figure 5-10 shows the geometry of the offsets; B_2 is the barycenter of an equilateral triangle formed by B_0, B_1, and B_3.

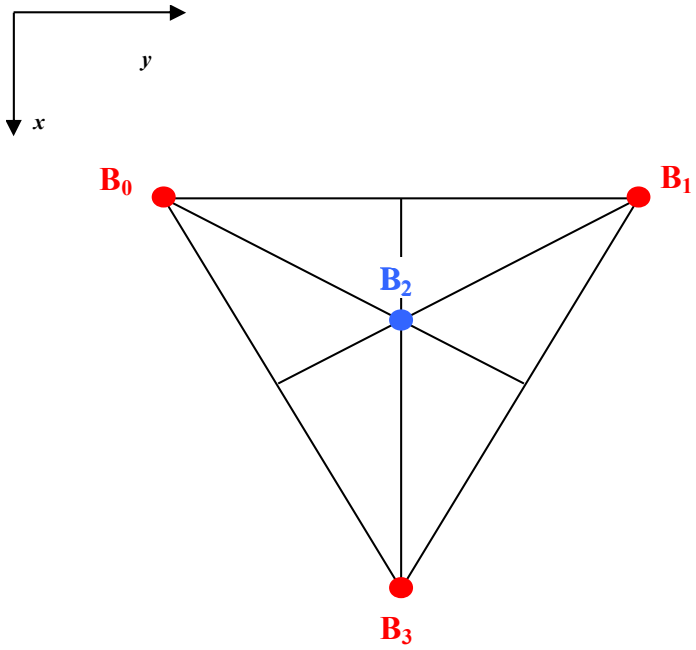


Figure 5-10: Sub-Pixel Misregistration Geometry

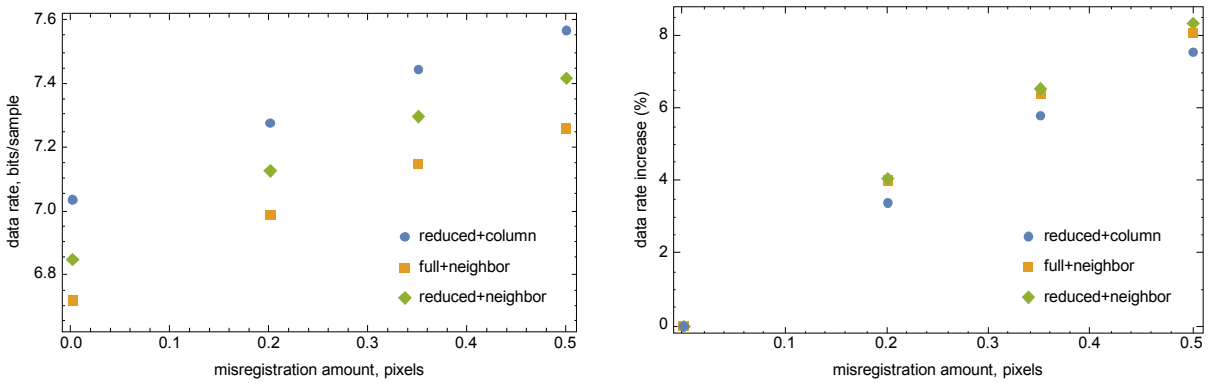


Figure 5-11: Impact of Sub-Pixel Misregistration on Lossless Compressed Data Rate for Pléiades Multispectral Images Using the Sample Adaptive Encoder

Figure 5-11 shows compressed data rate as a function of the misregistration amount, quantified as the distance between B_2 and any of the other three points drawn in figure 5-10. Sub-pixel misregistration reduces compression effectiveness by similar amounts for all predictor configurations tested. Not surprisingly, increasing amounts of sub-pixel misregistration yield higher reductions in compression performance.

5.6.2 PIXEL MISREGISTRATION

To illustrate the impact of more severe misregistration on compression performance, each band of the Pléiades sample image is translated by an integer amount between -1 and 1 in the x and y directions. Results are reported in table 5-1 using full compression mode with neighbor-oriented local sums and the sample-adaptive entropy coder. In this example, pixel misregistration yields an increase in compressed data rate as high as 0.91 bits/sample. Not surprisingly, these larger offsets yield noticeably higher compressed data rates than sub-pixel misregistration.

Table 5-1: Impact of Pixel Misregistration on Compressed Data Rate for Pléiades Images

				Difference with reference	
Case	**x-axis misregistration**	**y-axis misregistration**	**Data Rate (bits/sample)**	**Data Rate (bits/sample)**	**%**
1	[-1,-1,1,0]	[-1,-1,-1,0]	7.66	0.95	14.2
2	[0,0,0,1]	[0,1,0,1]	7.57	0.86	12.8
4	[-1,1,0,0]	[-1,-1,0,1]	7.66	0.95	14.2
5	[0,0,0,-1]	[1,-1,-1,1]	7.57	0.86	12.8
Mean			**7.62**	**0.91**	**13.5**
Reference	[0,0,0,0]	[0,0,0,0]	6.71	0	0

5.7 DECOMPRESSION

When the decompressor reconstructs sample $s_z(t)$ with value $s''_z(t)$ (defined in equation (48) of the Recommended Standard), Peak Absolute Error (PAE) is minimized, and reconstruction error is at most $m_z(t)$.

However, the standard imposes no requirements on how reconstructed sample values are to be selected by the decompressor. When compression is not lossless ($m_z(t) > 0$), selecting a reconstructed value other than $s''_z(t)$ may provide lower distortion when fidelity is assessed using another metric, such as RMSE.

As an example, figure 5-12 compares rate-distortion performance on the AVIRIS Maine image (compressed using $\Theta = 4$, $\phi_z = 0$, $\psi_z = 12$) when samples are reconstructed using $s_z''(t)$ to the performance obtained by using a Gaussian model to select reconstruction points.[6] In this example, the reconstruction strategy that minimizes PAE does not minimize RMSE. It should be noted that the same compressed image is used for both reconstruction methods; the improvement in RMSE performance did not require any changes to compression parameters.

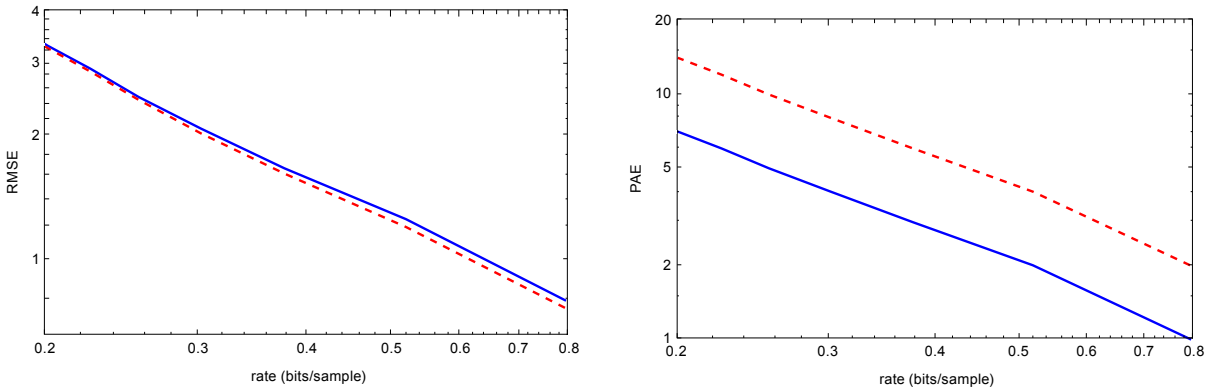


Figure 5-12: Compressed Data Rate versus Distortion When Samples Are Reconstructed to Minimize PAE (Blue) and Using a Gaussian Centroid Model (Red)

5.8 HARDWARE IMPLEMENTATION

5.8.1 INTRODUCTION

The Recommended Standard defines compression calculations in a way that facilitates relatively low complexity FPGA and ASIC hardware implementations. The algorithm uses only integer arithmetic, and all mathematical operations can be implemented with fixed shifters, barrel shifters, multipliers, and adders. Integer divisions are necessary when near-lossless compression is performed (i.e., when $m_z(t) \neq 0$).

The Recommended Standard allows one to use image partitioning to increase throughput by operating parallel compressors, each working on a portion of the image to be compressed. The trade-off between partitioning strategies and compression performance is discussed in subsection 5.3.2.

[6] Specifically, prediction error is assumed to be a zero-mean IID Gaussian, with variance in each band estimated by counting the fraction of samples in a band having nonzero quantizer index. The reconstruction point (for samples with nonzero quantizer index) is determined by computing the centroid of the quantizer bin, rounded to the nearest integer.

5.8.2 IMPLEMENTATIONS

Aranki et al. have implemented the FL compressor, on which issue 1 (reference [8]) of the present Recommended Standard is based, on Xilinx Virtex-4 (references [39] and [40]) and Virtex-5 FPGAs (reference [41]). The Intellectual Property (IP) core 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 Msamples/sec. The implementation has a low utilization of the Xilinx Virtex-5 XC5VSX50T (38 percent of Slice LUTs).

A low-complexity implementation of the Recommended Standard was developed by Santos et al. (reference [42]), implementing lossless compression and the sample-adaptive entropy coder. Synthesis results were obtained on a space-qualified RTAX1000S and a Virtex-5 XC5VFX130. The compression IP core has an occupancy of 34 percent of the RTAX1000S with a maximum frequency of 43 MHz, producing a throughput of 4 MSamples/sec. When synthesized in a Virtex-5 XC5VFX130, device utilization is very low (2 percent of slice LUTs) and the maximum frequency is 134 MHz, yielding a throughput of 11.3 MSamples/sec for a configuration with $P = 3$ and BSQ encoding order. Reference [43] presents a parallel implementation of the same core, devised by implementing several cores on a scalable onboard processor and partitioning the image to distribute the work among the cores, achieving a ten-fold acceleration when 16 cores are used compared to single-core execution. A more comprehensive implementation is presented in reference [44], which includes both sample-adaptive and block-adaptive entropy coding options. The implementation is fully parametrizable, allowing users to adjust compression parameters to control the compromise between complexity and compression efficiency to meet mission demands. The most complex design options utilize 4% of the LUTs in the Virtex-5 XC5VFX130, and the highest performance is achieved with BIP encoding order, reaching a throughput of 127 Msamples/sec.

A very high data rate hardware accelerator for the issue 1 compressor, using the sample-adaptive encoder, was implemented by Tsigkanos et al. (reference [45]). The architecture, based on the principles of C-slow retiming, exploits the inherent task-level parallelism of the algorithm under BIP encoding and implements a reconfigurable fine-grained pipeline in critical feedback loops, achieving a maximum throughput of 213 MSamples/s (3.3Gbps at bit depth $D = 16$) using 11% of the LUTs and 27% of the Block Random Access Memory (BRAM) units of the Virtex-5 XC5VFX130-1 FPGA resources when configured for a typical hyperspectral image (e.g., AVIRIS) using $P = 3$ prediction bands. The implementation is configurable at compile-time, setting the limits of allowable run-time configuration, which enables tailoring the IP core resources and attainable throughput performance to meet mission requirements.

The Fast Lossless Extended algorithm (FLEX), on which the issue 2 of the Recommended Standard is based, has been implemented by Keymeulen et al. (reference [46]). This implementation quantization to provide near-lossless compression and encodes the mapped quantizer indices using the hybrid entropy coder. The implementation targets the Virtex-7 XC7VX690T and Virtex-5 FPGAs FX130TFF1738. Results on the Virtex-5 show a device utilization of 20 percent of slice LUTs, and a maximum frequency of 168.8 MHz, yielding 24 MSamples/sec throughput. Higher throughput can be achieved if several instances of the compression core are used in parallel.

5.8.3 IMPLEMENTATION CONSIDERATIONS

5.8.3.1 General

When developing a hardware implementation of the Recommended Standard it is necessary to consider the factors that contribute the most to complexity. The complexity of the mathematical operations in the Recommended Standard is generally low; thus the following discussion focuses on the main constraints in terms of storage requirements and throughput limitations caused by data dependencies.

General notes are provided here regarding the complexity of prospective hardware implementations of the Recommended Standard; 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 an implementation using sample-adaptive coding and lossless compression is shown in figure 5-13, in which the blocks represent the main stages of the Recommended Standard.

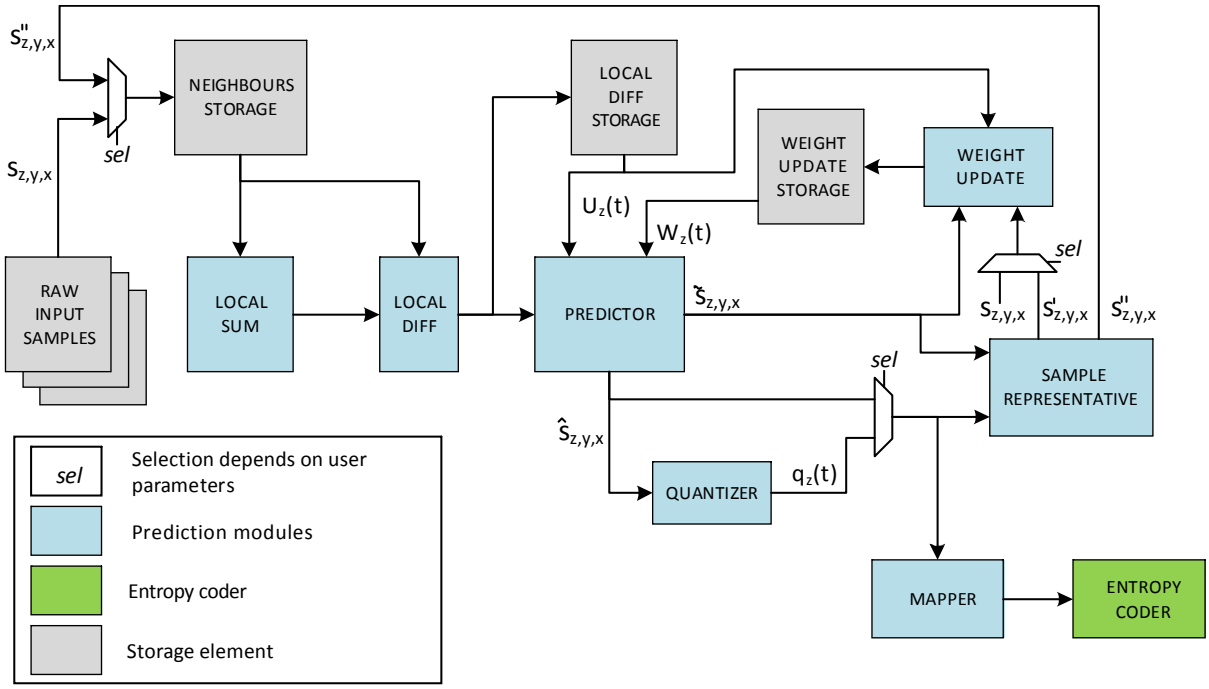


Figure 5-13: Simplified Schematic of a Lossless Compression Implementation, Highlighting User-Selectable Options that Affect Complexity

5.8.3.2 Complexity trade-offs for lossless compression

Near-lossless compression is achieved with a closed-loop quantization scheme, utilizing sample representative $s''(t)$ in place of the original sample $s(t)$ in the prediction calculation. When only lossless compression is desired, the quantization loop is not needed, and the sample representative calculation is eliminated (see 3.4), which simplifies the implementation and can improve throughput.

Specifically, when only lossless compression is needed, complexity can be reduced by using the following compression settings:

– Setting the quantizer fidelity control method to lossless ($m_z(t) = 0$) reduces complexity because the quantization calculation is eliminated, which eliminates the need for the integer division operations in quantization and mapping. This allows weight update calculations to begin earlier, thus increasing throughput.

– Hardware complexity is reduced by setting sample representative parameters to $\phi_z = \psi_z = 0$ for all z, which eliminates the sample representative calculation. This may increase throughput by removing data dependencies. The prediction of a new sample can begin before the sample representatives are available, as only the original previously compressed neighboring samples are needed. It should be noted that the compressed data rate may be lower when nonzero values of ϕ_z and ψ_z are used (see 3.4).

Storage requirements are similar for lossless and near-lossless compression, and depend mostly on the sample encoding order and the choice of parameters that involve storing tables with constants. The header is more compact when compression is lossless.

5.8.3.3 Local Sum and Local Differences Calculation

Prediction is causal; that is, only previously processed neighboring samples are needed to compute the local sum and local differences. To avoid the impact on latency of having to read the neighboring samples needed to perform a prediction calculation, one could temporarily arrange the previously processed samples in a storage component, such as a RAM memory or a First-In-First-Out (FIFO), as shown in references [39] and [42]. The component would store the sample representatives $s''_z(t)$, which are simply equal to the original sample when $\phi_z = \psi_z = 0$.

When ϕ_z and ψ_z are not both zero, data dependencies necessitate waiting until the sample representative is available to predict a new sample. This limits the possible pipelining of the compression of successive samples, and bounds the achievable throughput, especially for BSQ and BIL processing orders. To reduce this impact, issue 2 of the Recommended Standard introduces narrow local sums which, unlike the wide local sums, depend on the sample representative in the previous band, $s''_{z-1,y,x-1}$, instead of the one in the current band $s''_{z,y,x-1}$ to predict sample $s_{z,y,x}$. Provided that there are enough samples in the x-dimension, $s''_{z-1,y,x-1}$ can be computed several clock cycles before it is required, making it available by the time prediction of $s_{z,y,x}$ begins. Local sums can be narrow or wide, and either neighbor-oriented or column-oriented, leading to four possible combinations.

5.8.3.4 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 (36) of reference [1]):

$$\hat{d}_z(t) = \mathbf{W}_z^T(t)\mathbf{U}_z(t). \tag{27}$$

The main concern here 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 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 and BIP orders.

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

- **Storage:** It is necessary to store the local difference vectors for samples in a given band in a way that allows them to be used for prediction in the next band. This means storing the following set of vectors: $\boldsymbol{U}_z(0),...,U_z(t),...,U_z(N_y \times N_x - 1)$, 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, $s_z(t+1)$, because the updated weight vector $W_z(t+1)$ has to be available to calculate $\hat{d}_z(t+1)$. Consequently, 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. Additionally, when compression is not lossless, another data dependency arises: sample representative $s_z''(\mathrm{t})$ is needed to calculate the local sum $\sigma_z(\mathrm{t}+1)$. This limitation may be avoided by using narrow column-oriented local sums.

Under BIP processing order the following constraints are observed:

- **Storage:** Following prediction of 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 $s_z(t+1)$, the next sample in the same band. This suggests storing the updated weight vectors in all the bands: $\boldsymbol{W}_0(t),...,W_z(t),...,W_{N_z-1}(t)$.

– **Data dependencies:** For lossless compression, if the sample representative calculation is eliminated (by setting $\phi_z = \psi_z = 0$ for all z), it is not necessary to complete the prediction calculation for sample $s_z(t)$ before predicting 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 $s_z(t)$. When ψ_z are not both zero, sample representative $s''_z(\mathrm{t})$ must be computed before the central local difference of the next sample in BIP order, $\hat{d}_{z+1}(t)$, which can limit achievable throughput.

Estimating the amount of storage required by a specific hardware architecture is necessary to determine whether additional memory external to the FPGA is needed. Nevertheless, users may find solutions to cope with the aforementioned limitations. For example, under BSQ order, one could avoid storing the local differences if the compressor calculates all the elements of the local difference vector for the prediction of each sample. This approach requires the necessary neighboring samples (or sample representatives) to be available for the current sample to be compressed as well as those in the P preceding bands, and it also involves computing P additional local sums and local differences. This solution has been adopted for software and hardware implementations described in reference [42].

5.8.3.5 Quantization and mapping

When compression is not lossless ($m_z(t) \neq 0$), the quantization calculation requires the division of two integer variables to determine the signed quantized index $q_z(t)$ and the unsigned mapped quantized index $\delta_z(t)$. Compared to other integer arithmetic operations such as additions or multiplications, division is more complex, requiring more area when implemented on an FPGA and many cycles to be completed.

Several iterative algorithms and vendor IP cores exist to optimize the implementation of integer divisions. Alternatively, especially when high fidelity compression is desired, a reasonable approach to achieve high throughput compression hardware is to use an external quantizer followed by lossless compression. (See the discussion of companding in 4.2.7.1) This ‘pre-quantization’ approach eliminates the need for direct and inverse quantization inside the prediction feedback loop.

Figure 5-14 shows the compression performance of this approach on the *aviris_yellowstone_sc0_cal* image. Pre-quantization provides competitive results, particularly when compared to the use of reduced prediction mode and narrow column-oriented local sums, which is expected to provide the fastest near-lossless hardware implementation.

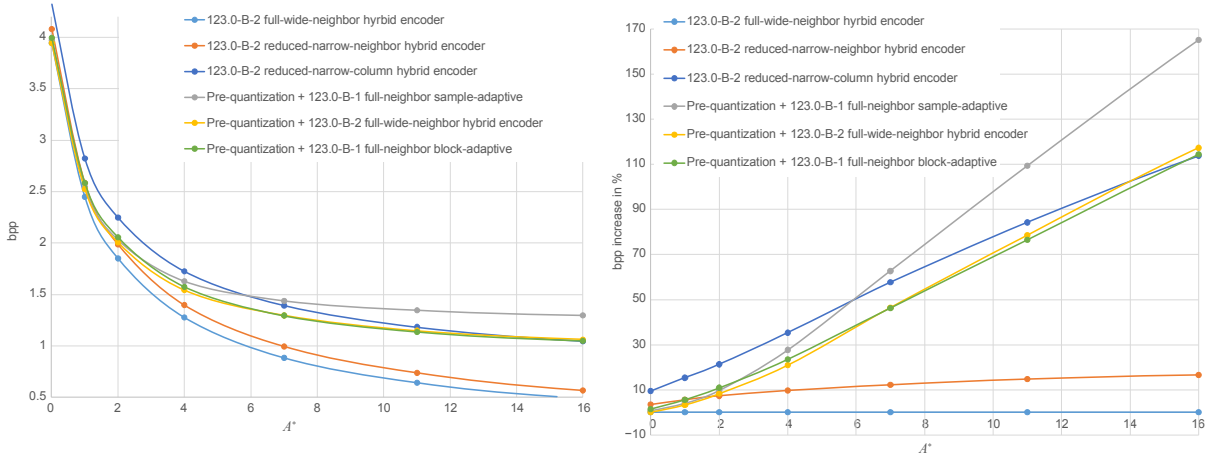


Figure 5-14: Comparison of Near-Lossless Compression As Specified in the Recommended Standard and the Use of an External Quantizer Prior to Lossless Compression

5.8.3.6 Entropy Coding

The block-adaptive coding approach requires the evaluation of the different encoding options for each block of J mapped quantizer indices. These evaluations require addition and comparison operations. The 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.

Encoding a mapped quantizer index using the sample-adaptive coding approach under BSQ order requires one previously processed mapped quantizer index, one accumulator, and one counter. Under BIL or BIP orders, the same elements are needed for each band in the image; that is, N_z mapped quantizer index, N_z accumulator, and N_z counter values are needed.

The hybrid entropy coder maintains essentially the same statistics as the sample-adaptive encoder, and thus requires storage of previously processed quantizer indices, accumulators, and counters. Nevertheless, the N_z accumulator values need to be stored for all sample encoding orders, as they need to be appended to generate the compressed image tail at the conclusion of encoding. The hybrid encoder also involves more decision-making operations than the sample-adaptive coder, and requires the storage of both code tables and flush tables.

Implementation considerations that may affect the choice between the three entropy coding approaches include:

– When sample-adaptive encoding is used, compression effectiveness does not depend on sample encoding order. When the hybrid entropy coder is used, assignment of mapped quantizer indices to component codes does not depend on sample encoding order, and thus compression effectiveness is essentially independent of encoding order. An implementer using the hybrid or 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.
- The hybrid and sample-adaptive coding approaches do not require the evaluation of multiple coding options; this may or may not provide a complexity advantage when compared to the block-adaptive coding approach.

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 quantizer indices 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 [47]. The hybrid entropy coder, like the sample-adaptive coder, does not involve the multiplication of integer variables, but it requires comparatively more operations (multiplications by constants, and barrel shifters). It also involves lookup operations to find the input codewords in the code tables.

6 PERFORMANCE

6.1 INTRODUCTION

This section presents compression performance results (compressed data rates in bits/sample and distortion metrics) for the Recommended Standard as well as some other compression approaches. Subsection 6.2 describes the compression methods evaluated. Subsections 6.3 and 6.4 provide lossless and lossy compression results, respectively, 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' refers 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, lossless 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 the wavelet-based JPEG2000 compressor, independently compressing image segments formed in this straightforward manner could cause compression efficiency to suffer noticeably. Instead, partitioning data in the transform domain achieves a similar degree of segmentation and allows 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. Subsection 6.2.5 describes 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 on board a spacecraft.

6.2 COMPRESSORS

6.2.1 OVERVIEW

This subsection lists the compressors used and describes the methods used to obtain the compression results.

6.2.2 CCSDS 123.0-B-2

For the Recommended Standard, when the block-adaptive coder is used, compression results are given for BIL and BIP encoding orders. Results are not included for BSQ encoding order because, as discussed in 4.3.4.2, compressed data rate under BSQ encoding order is nearly identical to that obtained under BIL order. Sample encoding order does not affect the data rates achieved by the sample-adaptive and hybrid encoders.

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.

Results for the Recommended Standard are produced using the compression settings indicated in annex C. Backwards-compatible and high-performance settings are used in subsections 6.3 and 6.4, respectively.

6.2.3 CCSDS-122.1-B-1

The CCSDS 122.1-B-1 compression standard (reference [48]) defines a three-dimensional image coder specifically designed to meet the constraints of spaceflight hardware. It extends the CCSDS 122.0-B-2 recommendation for two-dimensional image data (reference [50]) by defining spectral transforms to be used as a preprocessing stage. An extensive discussion of features and performance is included in references [49] and [51].

Results for CCSDS 122.1-B-1 were produced using the compression settings described in annex H of reference [51].

6.2.4 JPEG-LS

JPEG-LS (reference [52]) 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 [16]), while maintaining low complexity. JPEG-LS is based on the LOCO-I (Low Complexity Lossless Compression for Images) algorithm (reference [16]).

The JPEG-LS image compression standard (reference [52]) supports compression of three-dimensional ('multi-component') images such as multispectral and hyperspectral images. However, JPEG-LS is primarily designed to exploit two-dimensional image structure and takes almost no advantage of similarities between adjacent 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 Recommended Standard significantly exploits 3D structure by using a small three-dimensional neighborhood for prediction. For this reason, some ad hoc methods of applying single-component (2D) JPEG-LS compression to multispectral and hyperspectral images were considered.

The application of JPEG-LS 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.

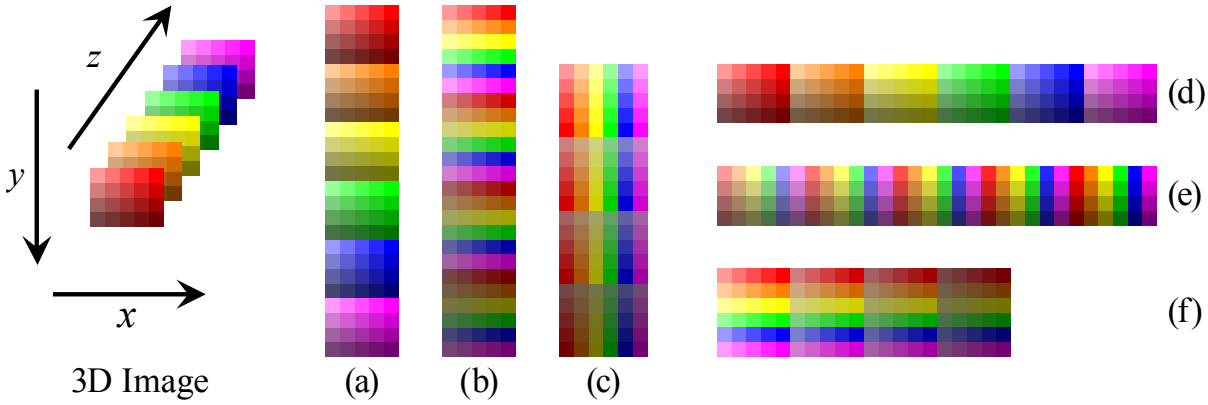


Figure 6-1: Examples of Different Methods of Forming a 2D Image by Flattening the Samples in a 3D Image

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 lossless compression 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.

Lossless compression results in 6.3 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.[7] 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.

Near-lossless compression results in 6.4 were produced using the JPEG-LS implementation available at https://github.com/thorfdbg/libjpeg/. Results were obtained via direct compression of spectral bands, which allows the maximum reconstruction error to be specified directly.

6.2.5 JPEG2000

The JPEG2000 image compression standard (reference [53]) defines a two-dimensional image coder based on wavelet transforms and bit plane encoding and provides a choice of integer and floating-point Discrete Wavelet Transforms (DWTs) so that effective lossy and lossless compression can be achieved. 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 compressed image size. 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. Hence, lossless compression can be achieved when a sufficiently large compressed image is allowed and the reversible wavelet transform is selected.

[7] http://www.stat.columbia.edu/~jakulin/jpeg-ls/

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. For lossy coding, Lagrangian rate-control can be performed, but this is not necessary for lossless coding. 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 [53]), and when a Pairwise Orthogonal Transform (POT) is employed (reference [54]).

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.

To provide a relevant comparison with other compressors applied to segmented images, JPEG2000 settings are selected to provide 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 that the communications data stream can be 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 = $(\min\{N/2^L, 64\}, 4096/\min\{N/2^L, 64\})$;
- Precinct size (level i = 0,1...) = $(N \cdot 2^{-i}, N_X)$.

These calculations are generic regardless of the number of components transformed or whether any spectral transform is used at all. 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.

Using the calculations above, the Kakadu software implementation[8] of JPEG2000 was used to produce experimental results for equivalent segment sizes of 32 frames (for hyperspectral images) or 256 frames (for multispectral images). For compression using the Kakadu software, the portions of the command line input that controls these settings are set respectively to:

```
Cblk={4,1024} Cprecincts={32,4096},{16,4096},{8,4096},{4,4096}
```

and

```
Cblk={8,512} Cprecincts={256,4096}, {128,4096},{64,4096},{32,4096},
{16,4096},{8,4096}.
```

All results are produced employing a reversible integer wavelet transform and a single quality layer (`Creversible=yes Clayers=1`).

6.2.6 LUT PREDICTOR

The LookUp-Table (LUT) compressor (reference [55]) 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 3.2.5) followed by lossless encoding using the sample-adaptive entropy coder. Reference [55] 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.

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

6.2.7 ESA COMPRESSOR

The lossless compressor proposed by ESA (reference [38]) targets very low complexity. The algorithm design is based on the block-based predictor first proposed in reference [56]. 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 using 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 [38].

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

6.3 LOSSLESS COMPRESSION RESULTS

6.3.1 FULL IMAGE VERSUS 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 issues like boundary handling or poorer adaptive prediction for predictive methods.

Figure 5-3 illustrates this reduction in lossless 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 D.

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		JPEG2000	
	Instrument	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	7.42	7.02
	AIRS	4.32	4.30	6.35	6.35	4.66	4.68	6.13	5.66	7.01	6.62
	CRISM-FRT	5.21	5.06	5.74	5.53	8.92	8.92	9.60	9.54	6.30	5.97
	CRISM-HRL	4.73	4.56	5.75	5.55	8.38	8.38	9.15	9.06	6.32	5.95
	CRISM-MSP	2.90	2.55	3.75	3.42	6.80	6.80	7.32	7.02	4.70	3.85
	M3-Global	2.37	2.14	4.93	4.83	6.45	6.45	7.30	7.20	5.40	5.10
	M3-Target	3.25	3.09	3.82	3.66	6.60	6.60	7.51	7.44	4.29	4.00
	Hyperion	4.37	4.30	5.09	5.02	5.52	5.52	6.16	6.08	5.39	5.14
	Hyperion flatfield	3.98	3.97	4.81	4.80	4.11	4.11	4.55	4.54	5.08	4.89
	SFSI	4.69	4.67	4.77	4.75	4.91	4.91	5.43	5.43	4.79	4.65
	SFSI_rmnoise	3.01	2.96	4.40	4.35	4.07	4.07	4.80	4.70	4.56	4.42
	AVIRIS(16-bit raw)	5.99	5.98	8.64	8.61	6.16	6.16	7.42	6.87	8.98	8.88
	AVIRIS(12-bit raw)	2.69	2.68	4.56	4.54	3.01	3.01	3.53	3.44	4.67	4.59
	AVIRIS(16-bit cal)	3.76	3.74	6.41	6.39	4.32	4.32	4.75	4.54	6.65	6.56
	CASI	5.03	5.02	6.79	6.77	5.10	5.10	5.96	5.65	7.09	6.97
Multispectral Images	MODIS-night	4.64	4.70	5.39	5.38	7.76	7.76	7.24	7.22	5.54	5.51
	MODIS-day	5.63	5.72	4.69	4.69	5.75	5.75	6.54	6.44	5.43	5.37
	MODIS-500m	7.19	7.20	7.63	7.62	8.36	8.36	8.88	8.88	7.80	7.77
	MODIS-250m	6.43	6.48	6.96	6.96	7.15	7.15	7.73	7.73	7.11	7.08
	MSG	3.40	3.39	3.50	3.50	4.11	4.11	5.07	5.06	3.53	3.50
	LANDSAT	3.37	3.37	3.70	3.70	3.91	3.91	4.32	4.32	3.90	3.87
	PLEIADES	7.32	7.32	7.85	7.84	7.95	7.95	8.74	8.65	8.25	8.15
	VEGETATION_level2	5.26	5.26	5.31	5.30	5.86	5.86	6.97	6.96	5.53	5.48
	VEGETATION_level3	5.04	5.04	5.27	5.26	5.55	5.55	6.77	6.77	5.46	5.42
	SPOT5	4.53	4.53	4.71	4.71	5.17	5.17	5.66	5.66	4.94	4.92

NOTE – Better performance is indicated in green.

Figure 6-2 focuses on the comparison of full versus segmented compression for the Recommended Standard. 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 full and segmented compression perform similarly and even a slight improvement sometimes occurs with segmented compression.

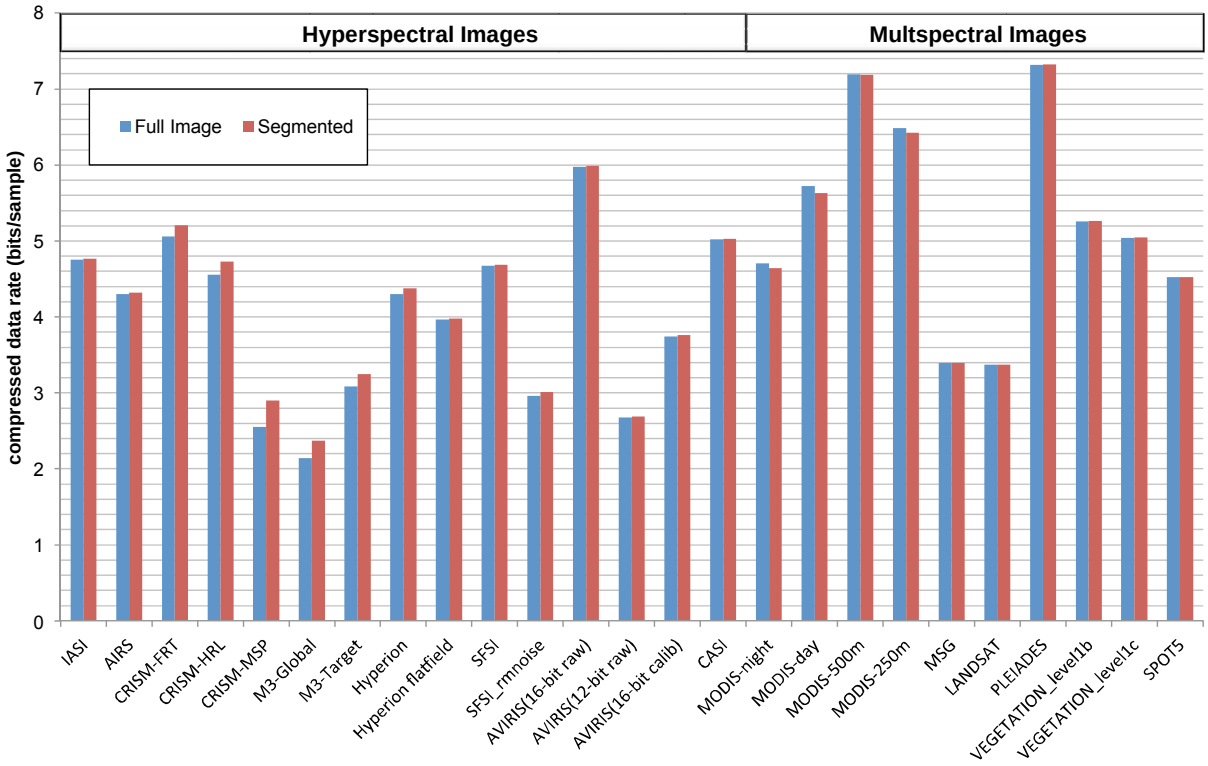


Figure 6-2: Comparison of Average Compressed Data Rates (in Bits/Sample) for the Recommended Standard for Full Image and Segmented Cases

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	JPEG2000		
			direct	differential			direct	+ IWT	+ POT
Hyperspectral Images	IASI	4.75	6.60	5.02	4.91	5.57	7.02	5.21	5.55
	AIRS	4.30	6.35	4.82	4.68	5.66	6.62	4.78	4.67
	CRISM-FRT	5.06	5.53	5.48	8.92	9.54	5.97	5.86	6.08
	CRISM-HRL	4.56	5.55	4.94	8.38	9.06	5.95	5.32	5.58
	CRISM-MSP	2.55	3.42	2.99	6.80	7.02	3.85	3.54	4.02
	M3-Global	2.14	4.83	2.56	6.45	7.20	5.10	3.13	3.56
	M3-Target	3.09	3.66	3.33	6.60	7.44	4.00	3.50	3.64
	Hyperion	4.30	5.02	4.59	5.52	6.08	5.14	4.52	4.58
	Hyperion flatfield	3.97	4.80	4.33	4.11	4.54	4.89	4.19	4.05
	SFSI	4.67	4.75	5.02	4.91	5.43	4.65	4.61	4.56
	SFSI_rmnoise	2.96	4.35	3.08	4.07	4.70	4.42	3.34	3.23
	AVIRIS(16-bit raw)	5.98	8.61	6.67	6.16	6.87	8.88	6.79	6.38
	AVIRIS(12-bit raw)	2.68	4.54	3.32	3.01	3.44	4.59	3.34	3.04
	AVIRIS(16-bit cal)	3.74	6.39	4.46	4.32	4.54	6.56	4.42	4.21
	CASI	5.02	6.77	5.33	5.10	5.65	6.97	5.28	5.37
Multispectral Images	MODIS-night	4.70	5.38	5.84	7.76	7.22	5.51	5.78	4.98
	MODIS-day	5.72	4.69	5.03	5.75	6.44	5.37	6.50	6.28
	MODIS-500m	7.20	7.62	6.64	8.36	8.88	7.77	7.18	6.95
	MODIS-250m	6.48	6.96	6.62	7.15	7.73	7.08	6.99	6.41
	MSG	3.39	3.50	3.83	4.11	5.06	3.50	3.87	3.70
	LANDSAT	3.37	3.70	3.62	3.91	4.32	3.87	3.67	3.60
	PLEIADES	7.32	7.84	7.31	7.95	8.65	8.15	7.66	7.92
	VEGETATION_level1	5.26	5.30	5.05	5.86	6.96	5.48	5.39	5.36
	VEGETATION_level1	5.04	5.26	5.02	5.55	6.77	5.42	5.31	5.25
	SPOT5	4.53	4.71	4.71	5.17	5.66	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 D.

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	JPEG2000		
			direct	differential			direct	+ IWT	+ POT
Hyperspectral Images Segment height = 32	IASI	4.77	6.61	5.03	4.91	5.72	7.42	5.64	6.02
	AIRS	4.32	6.35	4.85	4.68	6.13	7.01	5.19	5.14
	CRISM-FRT	5.21	5.74	5.73	8.92	9.60	6.30	6.22	6.40
	CRISM-HRL	4.73	5.75	5.21	8.38	9.15	6.32	5.75	5.98
	CRISM-MSP	2.90	3.75	3.47	6.80	7.32	4.70	4.45	4.95
	M3-Global	2.37	4.93	2.91	6.45	7.30	5.40	3.57	3.94
	M3-Target	3.25	3.82	3.56	6.60	7.51	4.29	3.85	3.98
	Hyperion	4.37	5.09	4.69	5.52	6.16	5.39	4.80	4.86
	Hyperion flatfield	3.98	4.81	4.34	4.11	4.55	5.08	4.40	4.27
	SFSI	4.69	4.77	5.03	4.91	5.43	4.79	4.72	4.67
	SFSI_rmnoise	3.01	4.40	3.16	4.07	4.80	4.56	3.51	3.38
	AVIRIS(16-bit raw)	5.99	8.64	6.71	6.16	7.42	8.98	6.87	6.46
	AVIRIS(12-bit raw)	2.69	4.56	3.34	3.01	3.53	4.67	3.41	3.13
	AVIRIS(16-bit cal)	3.76	6.41	4.49	4.32	4.75	6.65	4.50	4.30
	CASI	5.03	6.79	5.36	5.10	5.96	7.09	5.40	5.49
Multispectral Images Segment height = 256	MODIS-night	4.64	5.39	5.85	7.76	7.24	5.54	5.80	5.01
	MODIS-day	5.63	4.69	5.04	5.75	6.54	5.43	6.57	6.32
	MODIS-500m	7.19	7.63	6.66	8.36	8.88	7.80	7.16	6.95
	MODIS-250m	6.43	6.96	6.63	7.15	7.73	7.11	7.02	6.43
	MSG	3.40	3.50	3.83	4.11	5.07	3.53	3.90	3.72
	LANDSAT	3.37	3.70	3.62	3.91	4.32	3.90	3.69	3.62
	PLEIADES	7.32	7.85	7.33	7.95	8.74	8.25	7.75	8.01
	VEGETATION_level1b	5.26	5.31	5.06	5.86	6.97	5.53	5.45	5.41
	VEGETATION_level1c	5.04	5.27	5.03	5.55	6.77	5.46	5.34	5.28
	SPOT5	4.53	4.71	4.72	5.17	5.66	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 table 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.

6.4 LOSSY COMPRESSION RESULTS

A variety of metrics may be used to quantify the quality of a reconstructed image. For a decompressed image, let $\dot{s}_{z,y,x}$ denote the reconstructed value used to represent sample $s_{z,y,x}$. Mean Square Error (MSE) distortion between original and reconstructed images is defined as

$$\mathrm{MSE} = \frac{1}{N_\mathrm{Z} N_\mathrm{Y} N_\mathrm{X}} \sum_{z,y,x} \left(s_{z,y,x} - \dot{s}_{z,y,x}\right)^2.$$

A related metric, signal-to-noise ratio (SNR) fidelity, measured in dB, is

$$\mathrm{SNR} = 10 \log_{10} \left(\frac{\sum_{z,y,x} s_{z,y,x}^2}{\sum_{z,y,x} \left(s_{z,y,x} - \dot{s}_{z,y,x}\right)^2} \right) = 10 \log_{10} \left(\frac{\frac{1}{N_\mathrm{Z} N_\mathrm{Y} N_\mathrm{X}} \sum_{z,y,x} s_{z,y,x}^2}{\mathrm{MSE}} \right)$$

and Peak Absolute Error (PAE) distortion is

$$\mathrm{PAE} = \max_{z,y,x} \left| s_{z,y,x} - \dot{s}_{z,y,x} \right|.$$

For images reconstructed following compression using the Recommended Standard, we set $\dot{s}_{z,y,x} = s'_{z,y,x}$, that is, each sample is reconstructed using the center of the quantizer bin, for all results reported here.

It should be noted that these ‘fidelity’ metrics quantify the discrepancy between original and reconstructed images, which might not truly reflect the quality or scientific value of the reconstructed image. In particular, when the input image includes artifacts, fidelity metrics reward the accurate reproduction of such artifacts.

Similarly, differences in rate-distortion performance between two compressors operating at very low bit rates (very high distortion) may be of somewhat academic interest, as poor-quality images may be of little use to the end user (see 4.2.7.2).

As discussed in Section 2, the Recommended Standard is a ‘near-lossless’ compressor, thus it provides a user-specified guarantee on PAE. Specifically, when absolute error limits are used, $\mathrm{PAE} \leq \max_z a_z$. JPEG-LS is also a near-lossless compressor and provides a similar capability. By contrast, the transform-based compressors are designed to reduce distortion in an average sense, rather than limit it on a per-sample basis. Figure 6-3 illustrates error histograms for reconstruction of a test image compressed to the same bit rate, and achieving nearly identical SNR, using the Recommended Standard and the JPEG2000 compressors.

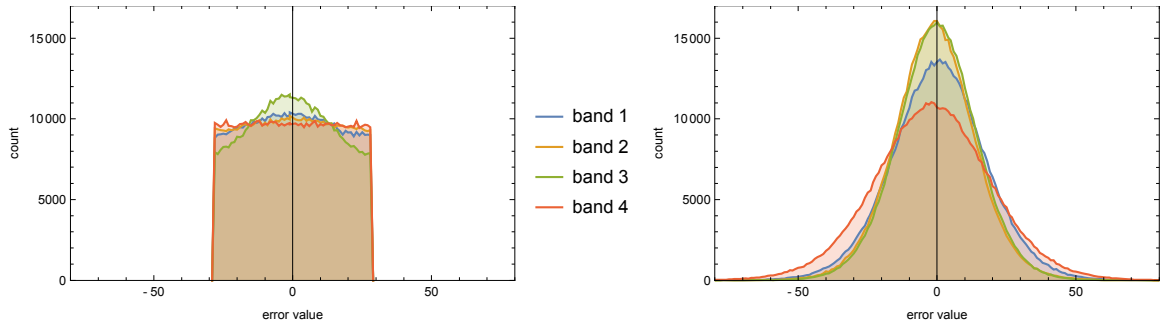


Figure 6-3: Histograms of Error Values in the Pléiades Test Image Compressed to 2 Bits/Sample (Left) Using the Recommended Standard (at $A^* = 28$, Yielding SNR 37.4 dB), (Right) Using JPEG2000 (Yielding SNR 37.7 dB)

For the Recommended Standard, when fidelity is controlled using absolute error limits and samples are reconstructed at the center of the quantizer bin, the error $\dot{s}_{z,y,x} - s'_{z,y,x}$ for any sample in band z must be an integer in the range $[-a_z, a_z]$. If errors in the band are uniformly distributed, then the MSE for the band is $a_z(a_z + 1)/3$. This turns out to be a good approximation for bands compressed to high data rates. Figure 6-4 shows band compressed data rate versus normalized MSE (MSE divided by $a_z(a_z + 1)/3$) for two of the test images at several different values of A^*.

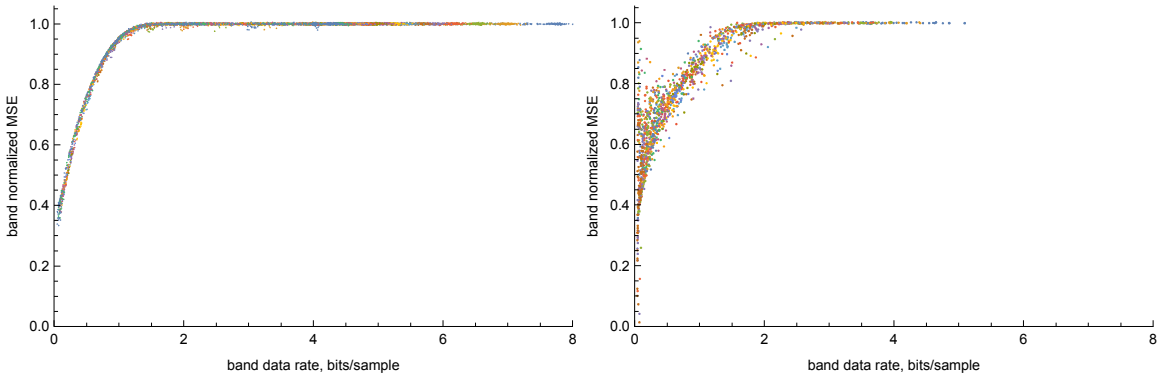


Figure 6-4: Normalized MSE versus Data Rate for Each Band of AVIRIS-NG (Left) and CASI (Right) Test Images, Compressed at Several Different Values of A^*

Figure 6-5 shows rate-distortion performance using the PAE metric on several test images in the corpus. For the images examined, the Recommended Standard, which is designed to meet a user-specified PAE constraint, provides better rate-PAE performance than the other compressors evaluated.[9]

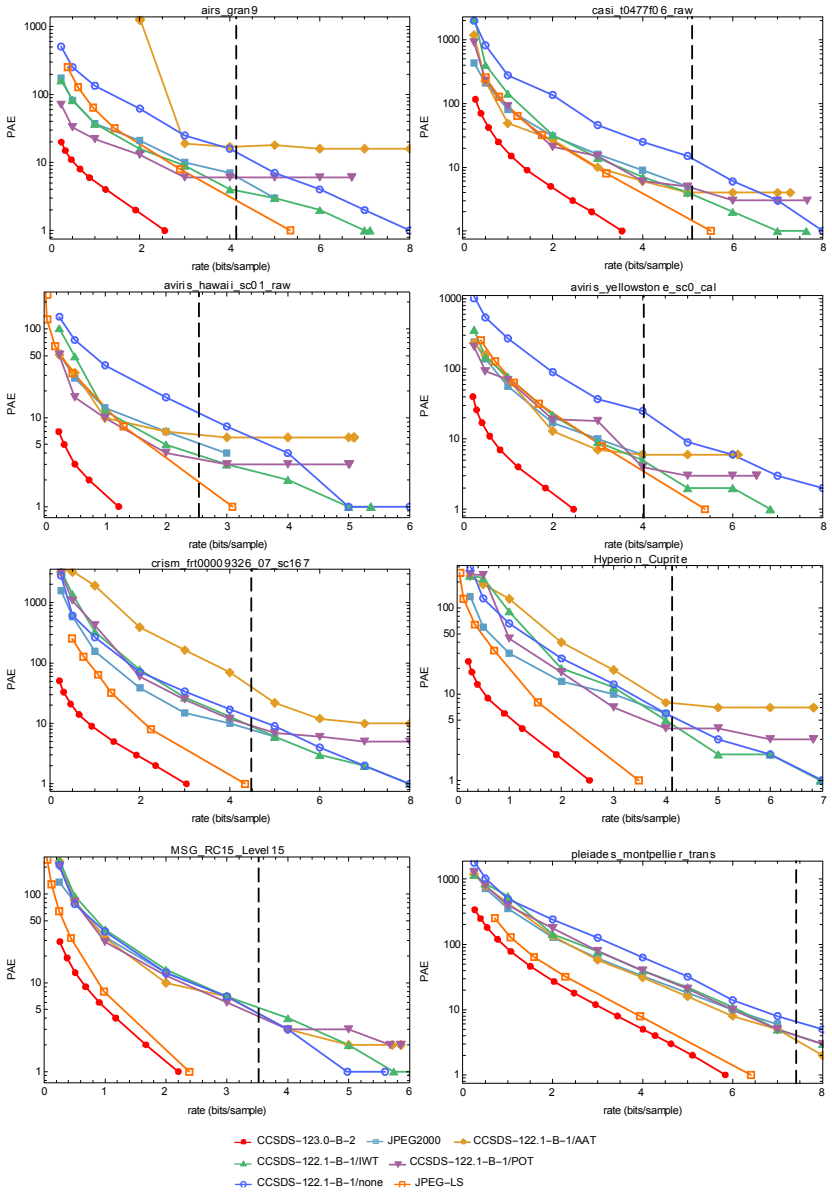


Figure 6-5: Compressed Data Rate versus PAE Performance of Different Compressors

NOTE – Each vertical dashed line indicates the rate at which the Recommended Standard achieves lossless compression (PAE = 0).

[9] The Arbitrary Affine Transform (AAT) option specified in reference [48] is not recommended for images with a large number of spectral bands because of the high relative cost of side information encoded (see subsection 6.3 of reference [51]). Thus the poor performance on the AIRS image (which has 1501 bands) is not surprising.

Figure 6-6 shows rate-SNR performance on several test images in the corpus. For the Recommended Standard, several rate-SNR points are obtained by varying the absolute error limit constant A^*. To obtain points at bit rates higher than that achieved at $A^* = 1$, we use band-dependent absolute error limits, with $a_z = 0$ for the first few bands, and $a_z = 1$ for all subsequent bands; the resulting performance is shown as a dashed red curve in the figure.

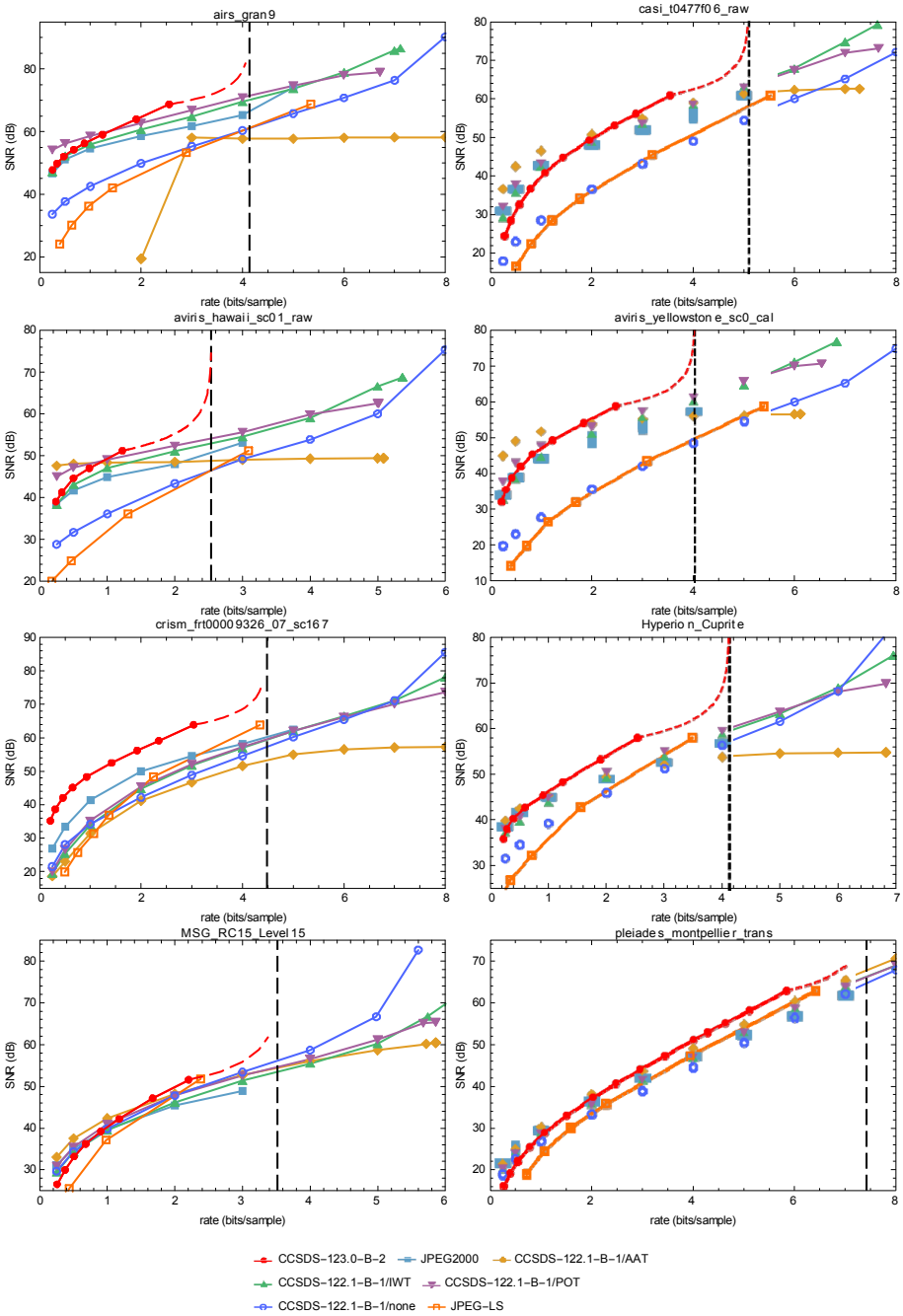


Figure 6-6: Compressed Data Rate versus SNR Performance of Different Compressors

NOTE – Each vertical dashed line indicates the rate at which the Recommended Standard achieves lossless compression.

Comparing the rate versus SNR performance of the Recommended Standard with the transform-based approach used by CCSDS-122.1-B and JPEG2000, it appears that a transform method has a higher chance of outperforming the predictive approach used by the Recommended Standard when:

a) compressed data rate is relatively low (not close to lossless);

b) the source image is relatively free from streaking artifacts;

c) spectral bands are more highly correlated (generally, more closely spaced in wavelength);

d) there are a large number of bands for the transform to exploit.

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	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
AVIRIS-NG	raw	16	432	640	512×4
AVIRIS-NG	calibrated	16	432	598	512×4
M3	target, raw	12	260	640	512×3
M3	global, raw	12	86	320	512×2
M3	radiance	16	85	304	512×4
Hyperion	raw	12	242	256	{1024, 3187, 3176, 3242}
Hyperion	flatfield-corrected	13	242	256	1024
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
HICO	calibrated	16	{128×2, 87×3}	{512×2, 500×3}	2000×5
CASI	raw	12	72	{405, 406}	{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
Pléiades	HR, simulated	12	4	{224×2, 296×4}	{2456, 3928, 2448×4}
Vegetation	raw	10	4	1728	{10080×2, 10193×2}
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), AVIRIS Next Generation (AVIRIS-NG), Hyperspectral Imager for the Coastal Ocean (HICO), Compact Airborne Spectrographic Imager (CASI), Moderate Resolution Imaging Spectroradiometer (MODIS), Meteosat Second Generation (MSG), Landsat, Pléiades 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 https://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 on board the Mars Reconnaissance Orbiter spacecraft by spatial and spectral binning of the samples from the imager. Similarly, M3 'global' images are produced on board 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:

https://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 the weight vector as specified in subsection 4.10.3 of reference [1]);
- result in overflow of the register used in the prediction calculation (thus helping to confirm correct calculation of equation (37) 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 $z = 1,2., \ldots, 15$; (c) last band ($z = 16$).

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

B3 TEST VECTORS

Test vectors for the Recommended Standard are available at:

https://cwe.ccsds.org/sls/docs/sls-dc/123.0-B-Info/TestData/TestVectors-B2

Fuzzing techniques have been employed on a feature-complete implementation of this recommendation to produce three exhaustive test sets. These test sets thoroughly exhaust all compression options and trigger features (such as clipping) that might occur extremely rarely for natural images using common combinations of compression parameters.

Test Set 1 consists of 7,946 test cases, each one using a synthetic input image produced via fuzzing, thoroughly exercising compression options. Each input image in Test Set 1 is fairly small. The largest has just over 130,000 samples, and half of them have fewer than 1300 samples. The total data volume of source images for Test Set 1 is 145 MBytes, and the total compressed image data volume is 1.5 MBytes. Test Set 1 includes 426 cases using supplementary information tables, exercising all table types, and 766 cases using side information.

Test Set 2 consists of 17,401 test cases using the test images described in annex A. Each source image is used in at least 190 test cases with different parameter combinations yielding total input data volume of 2.1 TBytes. The total data volume of the compressed images produced from Test Set 2 is 902 GBytes. None of the Test 2 cases made use of supplementary information tables or side information.

Test Set 3 consists of 9554 sample test files that are NOT compliant with CCSDS-123.0-B-2.

B4 HYBRID ENTROPY CODER TABLES

The low-entropy code tables provided in annex B of reference [1] are available in a machine-readable format at the following location.

https://cwe.ccsds.org/sls/docs/SLS-DC/123.0-B-Info/CCSDS_123.0-B-2_Hybrid_code_tables.tar.gz

ANNEX C

COMPRESSION SETTINGS USED FOR EXPERIMENTS

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

Default prediction settings are given in table C-1. Except where noted otherwise, backwards-compatible and high-performance compression results made use of the sample-adaptive entropy coder with settings given in table C-2, and the hybrid entropy coder with settings given in table C-3. In cases in which block-adaptive coding was used, the default block size used was $J = 64$, and the reference sample interval was $r = 4096$.

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.

When compression is not lossless, reconstructed image fidelity is controlled using band-independent absolute error limits via the value of the integer absolute error limit constant parameter A^*. It should be noted that setting $A^* = 0$ yields lossless compression and is equivalent to selecting the fidelity control method to be lossless, apart from some minor differences in the compressed image header. Relative error limits are not used, and periodic error limit updating is not used; that is, a fixed value of A^* is used for the entire image.

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		Wide column-oriented for AVIRIS-NG, CRISM, Hyperion raw, M3, MODIS, SFSI rad; wide neighbor-oriented otherwise	Identifies neighborhood used to calculate local sums
Prediction Mode		reduced for AIRS, AVIRIS-NG, CASI raw, CRISM, HICO, 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

Name	Symbol	Setting	Description
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	$v_{\max}$	$7-\phi_z$	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
Weight Update Scaling Exponent offsets	$\left\{\zeta_z^{(i)}, \zeta_z^*\right\}$	All are set to 0	Offsets the exponent of each predictor weight update calculation
Periodic error limit updating		Disabled	Enables or disables periodic updates of error limits
Sample representative resolution	Θ	3	Determines the precision of sample representative calculations
Sample representative offset	$\{\psi_z\}$	7	Controls the offset from the center of the quantizer bin in the sample representative calculation.
Band-varying offset		Disabled (all ψ_z are equal)	Enables the use of different values for each element in ψ_z
Sample representative damping	$\{\phi_z\}$	5 for AVIRIS-NG, HICO, SFSI; 3 for AIRS, AVIRIS 12-bit raw, CASI, CRISM, Hyperion, M3 Target; 0 otherwise	Controls the tradeoff between predicted sample value and offset bin center in the sample representative calculation
Band-varying damping		Disabled (all ϕ_z are equal)	Enables the use of different values for each element in ϕ_z

Table C-2: Sample-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

Table C-3: Hybrid 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
Initial high-resolution accumulator value	$\{\tilde{\Sigma}_z(0)\}$		Sets an initial accumulator value for each band

ANNEX D

COMPRESSION RESULTS

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

Tables D-5 and D-6 compare lossless compression performance of high-performance and backwards-compatible settings for hyperspectral and multispectral images, respectively.

Table D-1: Compressed Data Rates (in Bits/Sample) for Lossless Full Image Compression of Hyperspectral Images Using Backwards-Compatible Compression Settings

		CCSDS-123.0-B										JPEG-LS		ESA + sample-adaptive coding	LUT + sample-adaptive coding	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
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	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	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	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	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***	***7.02***	***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	6.69	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	6.44	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	7.15	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	6.04	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	6.61	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	7.01	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	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	6.74	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	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	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***	***6.62***	***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	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	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	6.6	6.39	6.67
CRISM-FRT	frt0000065e6_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	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	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	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	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	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	6.19	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***	***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	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	6.34	4.94	5.32
CRISM-HRL	hrl000089fd_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	5.5	5.35	5.53
CRISM-HRL	hrl0000648f_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	6.15	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***	***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	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	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	4.03	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	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	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	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	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***	***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	5.14	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	5.05	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***	***5.10***	***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	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	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	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***	***4.00***	***3.50***	***3.64***

		CCSDS-123.0-B										JPEG-LS		ESA + sample-adaptive coding	LUT + sample-adaptive coding	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
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	5.07	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	5.08	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	5.31	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	5.1	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***	***5.14***	***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	4.89	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	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	4.42	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	9.46	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	9.19	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	7.45	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	8.74	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	9.58	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***	***8.88***	***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	4.62	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	4.55	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***	***4.59***	***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	7.14	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	6.86	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	5.15	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	6.39	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	7.24	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***	***6.56***	***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	6.6	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	7.34	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***	***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	9.46	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	10.7	8.7	8.8

Table D-2: Compressed Data Rates (in Bits/Sample) for Lossless Full Image Compression of Multispectral Images Using Backwards-Compatible Compression Settings

		CCSDS-123.0-B										JPEG-LS		ESA + sample-adaptive coding	LUT + sample-adaptive coding	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
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	5.13	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	6.09	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	5.1	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	6.23	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	5.02	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***	***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	3.31	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	5.74	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	6.15	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	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	5.68	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***	***5.37***	***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	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	8.09	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	7.25	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	8.27	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	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***	***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	6.94	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	7.34	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	6.7	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	7.69	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	6.75	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***	***7.08***	***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	3.43	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	3.69	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	3.38	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***	***3.50***	***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	4.27	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	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	4.34	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***	***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	8.3	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	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***	***8.15***	***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	7.53	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	7.53	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	7.53	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	7.53	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	5.46	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	5.5	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***	***5.48***	***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	5.4	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	5.44	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***	***5.42***	***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	5.61	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	4.58	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	4.57	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***	***4.92***	***5.03***	***4.61***

Table D-3: Compressed Data Rate (in Bits/Sample) for Segmented Lossless Compression of Hyperspectral Images Using Backwards-Compatible Compression Settings

		CCSDS-123.0-B										JPEG-LS		ESA + sample-adaptive coding	LUT + sample-adaptive coding	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
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	**7.01**	**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	**8.17**	**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	**7.14**	**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	**7.35**	**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***	***7.42***	***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	**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	**6.82**	**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	**7.53**	**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	**6.42**	**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	**6.98**	**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	**7.39**	**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	**6.17**	**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	**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	**7.21**	**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	**7.34**	**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***	***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	**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	**6.23**	**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	**6.93**	**6.75**	**6.99**
CRISM-FRT	frt000065e6_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	**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	**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	**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	**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	**6.07**	**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	**6.48**	**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***	***6.30***	***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	**6.23**	**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	**6.65**	**5.36**	**5.69**
CRISM-HRL	hrl000089fd_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	**5.92**	**5.79**	**5.95**
CRISM-HRL	hrl0000648f_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	**6.49**	**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***	***6.32***	***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	**4.22**	**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	**4.46**	**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	**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	**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	**4.58**	**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	**5.14**	**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	**5.72**	**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***	***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	**5.44**	**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	**5.36**	**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***	***5.40***	***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	**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	**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	**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***	***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	**5.33**	**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	**5.34**	**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	**5.55**	**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	**5.34**	**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***	***5.39***	***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	**5.08**	**4.4**	**4.27**

Instrument	scene	CCSDS-123.0-B										JPEG-LS		ESA + sample-adaptive coding	LUT + sample-adaptive coding	JPEG2000		
		defaults	sample-adaptive			block-adaptive BIL			block-adaptive BIP									
			reduced+ column	full+ neighbor	reduced+ neighbor	reduced+ column	full+ neighbor	reduced+ neighbor	reduced+ column	full+ neighbor	reduced+ neighbor	direct	differential			direct	+ IWT	+ POT
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	**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	**4.56**	**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	**9.54**	**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	**9.27**	**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	**7.57**	**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	**8.83**	**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	**9.67**	**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***	***8.98***	***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	**4.7**	**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	**4.64**	**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***	***4.67***	***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	**7.22**	**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	**6.95**	**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	**5.28**	**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	**6.47**	**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	**7.32**	**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***	***6.65***	***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	**6.71**	**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	**7.46**	**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***	***7.09***	***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	**9.57**	**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	**10.82**	**8.81**	**8.92**

Table D-4: Compressed Data Rate (in Bits/Sample) for Segmented Lossless Compression of Multispectral Images Using Backwards-Compatible Compression Settings

		CCSDS-123.0-B										JPEG-LS		ESA + sample-adaptive coding	LUT + sample-adaptive coding	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
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	5.14	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	6.12	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	5.13	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	6.26	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	5.05	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*	*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	3.35	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	5.8	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	6.22	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	6.01	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	5.76	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*	*5.43*	*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	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	8.13	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	7.28	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	8.29	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	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*	*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	6.97	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	7.37	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	6.73	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	7.71	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	6.77	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*	*7.11*	*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	3.46	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	3.72	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	3.41	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*	*3.53*	*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	4.29	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	3.05	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	4.35	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*	*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	8.4	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	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*	*8.25*	*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	7.6	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	7.6	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	7.6	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	7.6	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	5.51	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	5.55	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*	*5.53*	*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	5.24
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	5.36	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*	*5.34*	*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	5.63	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	4.61	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	4.59	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*	*4.94*	*5.06*	*4.64*

Table D-5: Compressed Data Rates (in Bits/Sample) for Full Image Compression of Hyperspectral Images Using High-Performance and Backwards-Compatible Settings

instrument	scene	high-performance	backwards-compatible
IASI	desert	4.72	4.7
IASI	ecosse	4.78	4.91
IASI	polenord	4.59	4.64
IASI	tropocean	4.76	4.77
IASI	**average**	**4.71**	**4.75**
AIRS	9	4.14	4.22
AIRS	16	4.11	4.2
AIRS	60	4.35	4.38
AIRS	82	4	4.13
AIRS	120	4.23	4.29
AIRS	126	4.37	4.4
AIRS	129	4.02	4.14
AIRS	151	4.38	4.41
AIRS	182	4.4	4.43
AIRS	193	4.38	4.42
AIRS	**average**	**4.24**	**4.3**
CRISM-FRT	frt00006d1c_07_sc164	4.44	4.61
CRISM-FRT	frt00010f86_07_sc167	5.12	5.28
CRISM-FRT	frt00013e49_07_sc166	5.3	5.46
CRISM-FRT	frt000065e6_07_sc164	4.65	4.83
CRISM-FRT	frt0000869b_07_sc164	4.89	5.06
CRISM-FRT	frt0001077d_07_sc166	5.22	5.39
CRISM-FRT	frt0001241d_07_sc164	5.19	5.35
CRISM-FRT	frt00008849_07_sc165	4.7	4.88
CRISM-FRT	frt00009326_07_sc167	4.48	4.65
CRISM-FRT	**average (FRT)**	**4.89**	**5.06**
CRISM-HRL	hrl0000ba9c_07_sc183	4.75	4.92
CRISM-HRL	hrl00004f38_07_sc181	4.08	4.26
CRISM-HRL	hrl000089fd_07_sc182	4.4	4.58
CRISM-HRL	hrl0000648f_07_sc182	4.28	4.46
CRISM-HRL	**average (HRL)**	**4.38**	**4.56**
CRISM-MSP	msp00003e34_01_sc214	2.34	2.37
CRISM-MSP	msp00003ea5_07_sc214	2.36	2.37
CRISM-MSP	msp00003fbc_03_sc214	2.4	2.42
CRISM-MSP	msp0000426b_01_sc214	2.29	2.33
CRISM-MSP	msp0000438d_03_sc214	2.35	2.36
CRISM-MSP	msp0001081c_07_sc214	3.33	3.38
CRISM-MSP	msp00004125_05_sc214	2.59	2.62
CRISM-MSP	**average (MSP)**	**2.52**	**2.55**
HICO	NewYork	4.98	5.11
HICO	Tiburon	4.95	5.05
HICO	**average**	**4.96**	**5.08**

instrument	scene	high-performance	backwards-compatible
M3-Global	globalA	2.24	2.17
M3-Global	globalB	2.19	2.12
M3-Global	**average (global)**	**2.21**	**2.14**
M3-Target	targetA	3.28	3.38
M3-Target	targetB	3.06	3.18
M3-Target	targetC	2.58	2.71
M3-Target	**average (target)**	**2.97**	**3.09**
M3-radiance	rad_A_int2	2.13	2.1
M3-radiance	rad_B_int2	2.09	2.04
M3-radiance	rad_C_int2	2.19	2.2
M3-radiance	rad_D_int2	2.16	2.16
M3-radiance	**average (radiance)**	**2.14**	**2.13**
Hyperion	ErtaAle	4.12	4.26
Hyperion	LakeMonona	4.25	4.39
Hyperion	MtStHelens	4.18	4.3
Hyperion	Cuprite	4.12	4.26
Hyperion	**average (raw)**	**4.17**	**4.3**
Hyperion	Cuprite_flatfield	4	4.01
SFSI	mantar_raw	4.48	4.67
SFSI	mantar_rad_rmnoise	3.16	2.96
AVIRIS	sc0_raw	6.16	6.19
AVIRIS	sc3_raw	6.02	6.06
AVIRIS	sc10_raw	5.59	5.58
AVIRIS	sc11_raw	5.84	5.83
AVIRIS	sc18_raw	6.1	6.21
AVIRIS	**average (16-bit raw)**	**5.94**	**5.98**
AVIRIS	hawaii	2.54	2.62
AVIRIS	maine	2.67	2.73
AVIRIS	**average (12-bit raw)**	**2.61**	**2.68**
AVIRIS	sc0_cal	4.03	3.96
AVIRIS	sc3_cal	3.89	3.83
AVIRIS	sc10_cal	3.42	3.37
AVIRIS	sc11_cal	3.7	3.64
AVIRIS	sc18_cal	3.97	3.91
AVIRIS	**average (16-bit cal)**	**3.8**	**3.74**
CASI	t0180f07	4.77	4.86
CASI	t0477f06	5.1	5.18
CASI	**average (raw)**	**4.93**	**5.02**
CASI	t0180f07_rad	7.55	7.58
CASI	t0477f06_rad_rss	8.24	8.3

Table D-6: Compressed Data Rates (in Bits/Sample) for Full Image Compression of Multispectral Images Using High-Performance and Backwards-Compatible Settings

instrument	scene	high-performance	backwards-compatible
MODIS	A2001123.0000_night	4.46	4.44
MODIS	A2001123.1630_night	5.13	5.1
MODIS	A2001222.0835_night	4.46	4.45
MODIS	A2001222.0840_night	5.17	5.15
MODIS	A2001222.1200_night	4.38	4.37
MODIS	**average**	**4.72**	**4.7**
MODIS	A2001123.0000_day	3.77	4.21
MODIS	A2001123.1630_day	5.68	6.04
MODIS	A2001222.0835_day	6.19	6.37
MODIS	A2001222.0840_day	5.87	6.22
MODIS	A2001222.1200_day	5.47	5.78
MODIS	**average**	**5.4**	**5.72**
MODIS	A2001123.0000_500m	6.82	6.82
MODIS	A2001123.1630_500m	7.51	7.52
MODIS	A2001222.0835_500m	6.91	6.92
MODIS	A2001222.0840_500m	7.65	7.68
MODIS	A2001222.1200_500m	7	7.04
MODIS	**average**	**7.18**	**7.2**
MODIS	A2001123.0000_250m	5.84	5.83
MODIS	A2001123.1630_250m	7.17	7.11
MODIS	A2001222.0835_250m	6.23	6.16
MODIS	A2001222.0840_250m	7.38	7.33
MODIS	A2001222.1200_250m	6.01	6
MODIS	**average**	**6.53**	**6.48**
MSG	3	3.3	3.34
MSG	15	3.52	3.54
MSG	31	3.27	3.3
MSG	**average**	**3.36**	**3.39**
LANDSAT	agriculture	3.58	3.6
LANDSAT	coast	2.73	2.74
LANDSAT	mountain	3.75	3.78
LANDSAT	**average**	**3.36**	**3.37**

instrument	scene	high-performance	backwards-compatible
PLEIADES	montp_trans	7.42	7.42
PLEIADES	perp_trans	7.2	7.21
PLEIADES	**average**	**7.31**	**7.32**
PLEIADES	montp_misreg0	6.71	6.71
PLEIADES	montp_misreg20	6.97	6.98
PLEIADES	montp_misreg35	7.13	7.14
PLEIADES	montp_misreg50	7.25	7.26
VEGETATION	1_1b	5.17	5.24
VEGETATION	2_1b	5.2	5.28
VEGETATION	**average (level 1b)**	**5.18**	**5.26**
VEGETATION	1_1c	5.03	5.02
VEGETATION	2_1c	5.07	5.06
VEGETATION	**average (level 1c)**	**5.05**	**5.04**
SPOT5	1	5.13	5.14
SPOT5	2	4.26	4.27
SPOT5	3	4.14	4.17
SPOT5	**average**	**4.51**	**4.53**

ANNEX E

ABBREVIATIONS AND ACRONYMS

AAT	Arbitrary Affine Transform
AIRS	Atmospheric Infrared Sounder
AVIRIS	Airborne Visible/Infrared Imaging Spectrometer
BI	band interleaved
BIL	band-interleaved lines
BIP	band-interleaved pixels
BRAM	block random access memory
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
FIFO	first in first out
FL	fast lossless
FLEX	FL extended
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
IP	Intellectual Property
ISO	International Organization for Standardization

ITU	International Telecommunication Union
IWT	integer wavelet transform
LMS	least mean square
LUT	lookup table
M3	Moon Mineralogy Mapper
MCT	multi-component transform
MODIS	Moderate Resolution Imaging Spectroradiometer
MSE	mean square error
MSG	Meteosat Second Generation
MSP	multispectral survey
PAE	peak absolute error
POT	pairwise orthogonal transform
PSNR	peak signal-to-noise ratio
PVN	packet version number
RMSE	root mean square error
RRMSE	relative RMSE
SFSI	SWIR Full Spectrum Imager
SNR	signal-to-noise ratio
SPOT5	Système Pour l'Observation de la Terre 5
SWIR	short wave infrared

ANNEX F

A PROOF

This annex provides a proof that satisfying inequality (66) of reference [1] with $k_z(t) = 1$ could only occur for an input image with bit depth $D = 3$ (see 3.3.5.3).

First, it may be noted that

$$\tilde{\Sigma}_z(t) \leq 4(2^D - 1)\Gamma(t), \text{ for all } t. \tag{28}$$

Inequality (28) is proved by induction. At $t = 0$, subsection 5.4.3.3.4.3 of [1] requires

$$\tilde{\Sigma}_z(0) < 2^{D+\gamma_0} = 2^D\Gamma(0) < 4(2^D - 1)\Gamma(0)$$

(this last inequality follows because the Recommended Standard requires $D \geq 2$). Thus inequality (28) is true at $t = 0$.

Supposing that inequality (28) is true at $t - 1$, that is, $\tilde{\Sigma}_z(t-1) \leq 4(2^D - 1)\Gamma(t-1)$, there are two cases to consider at t: if $\Gamma(t-1) = 2^{\gamma^*} - 1$ then $\tilde{\Sigma}_z$ and Γ are rescaled at t, otherwise they are not rescaled.

If rescaling does not occur at t, then $\Gamma(t) = \Gamma(t-1) + 1$, and since $\delta_z(t) \leq 2^D - 1$, it can be found that

$$\begin{aligned}\tilde{\Sigma}_z(t) = \tilde{\Sigma}_z(t-1) + 4\delta_z(t) &\leq 4(2^D - 1)\Gamma(t-1) + 4\delta_z(t)\\ &\leq 4(2^D - 1)\Gamma(t-1) + 4(2^D - 1) = 4(2^D - 1)(\Gamma(t-1) + 1)\\ &= 4(2^D - 1)\Gamma(t)\end{aligned}$$

If rescaling occurs at t, then $\Gamma(t-1) = 2^{\gamma^*} - 1$, $\Gamma(t) = 2^{\gamma^*-1}$, and

$$\begin{aligned}\tilde{\Sigma}_z(t) = \left\lfloor \frac{\tilde{\Sigma}_z(t-1) + 4\delta_z(t) + 1}{2} \right\rfloor &\leq \left\lfloor \frac{4(2^D - 1)\Gamma(t-1) + 4\delta_z(t) + 1}{2} \right\rfloor\\ &= 2(2^D - 1)\Gamma(t-1) + 2\delta_z(t) \leq 2(2^D - 1)\Gamma(t-1) + 2(2^D - 1)\\ &= 2(2^D - 1)(\Gamma(t-1) + 1) = 2(2^D - 1)\left(2^{\gamma^*} - 1 + 1\right) = 4(2^D - 1)\left(2^{\gamma^*-1}\right)\\ &= 4(2^D - 1)\Gamma(t)\end{aligned}$$

which completes the proof of inequality (28).

In fact, inequality (28) can be met with equality with a sufficiently long input sequence of maximum-valued mapped quantizer indices $\delta_z(t) = 2^D - 1$. (It should be noted that such a sequence is not possible unless $m_z(t) = 0$.)

Applying inequality (28) when $D = 2$ yields

$$\tilde{\Sigma}_z(t) \leq 12\Gamma(t) < \frac{T_0}{2^{14}}\Gamma(t).$$

And thus the condition specified in 5.4.3.3.5.1.4 of reference [1] for $\delta_z(t)$ to be encoded as a high-entropy mapped quantizer index cannot be met when $D = 2$ even if it were permitted by the Recommended Standard.

When $D = 3$, it *is* possible for $\tilde{\Sigma}_z(t)/\Gamma(t)$ to be large enough to cause high-entropy coding to be used if it were permitted.

A mapped quantizer index meets the high-entropy threshold if $\tilde{\Sigma}_z(t) \cdot 2^{14} \geq T_0 \cdot \Gamma(t)$ (see subsection 5.4.3.3.5.1.4 of reference [1]). Thus for any high-entropy mapped quantizer index,

$$\tilde{\Sigma}_z(t) \geq \frac{T_0}{2^{14}} \cdot \Gamma(t) = \frac{303336}{2^{14}} \cdot \Gamma(t) > 18 \cdot \Gamma(t).$$

So, considering inequality (66) reference [1], it is found that, if $k_z(t) = 2$, then

$$\Gamma(t)2^{k_z(t)+2} = 16\Gamma(t) < 18\Gamma(t) < \tilde{\Sigma}_z(t) < \tilde{\Sigma}_z(t) + \left\lfloor \frac{49}{2^5}\Gamma(t) \right\rfloor.$$

That is, for any high-entropy mapped quantizer index, inequality (66) in reference [1] is always satisfied for $k_z(t) = 2$. The equation is not always satisfied for $k_z(t) = 3$, (e.g., when $\Gamma(t) = 2000$ and $\tilde{\Sigma}_z(t) = 40000$).

Thus:

- When $D = 2$, no mapped quantizer indices will meet the high-entropy threshold.
- When $D = 3$, if the Recommended Standard allowed mapped quantizer indices to be classified as high-entropy, they would be encoded using $k_z(t) = 1$.
- When $D = 4$, all high-entropy mapped quantizer indices will be encoded using $k_z(t) = 2$.
- When $D > 4$, each high-entropy mapped quantizer index coding option is bounded by $k_z(t) \geq 2$.