Hydrol. Earth Syst. Sci. Discuss., 10, 2029–2065, 2013
www.hydrol-earth-syst-sci-discuss.net/10/2029/2013/
doi:10.5194/hessd-10-2029-2013

Hydrology and Earth System Sciences
Open Access
Discussions

Data compression to define information content of hydrological time series

S. V. Weijs[1], **N. van de Giesen**[2], **and M. B. Parlange**[1]

[1]Environmental fluid mechanics and Hydrology Lab, ENAC, Ecole Polytechnique Fédérale de Lausanne, Station 2, 1015 Lausanne, Switzerland
[2]Water resources management, Delft University of Technology, Stevinweg 1, P.O. Box 5048, 2600 GA, Delft, The Netherlands

Received: 31 January 2013 – Accepted: 6 February 2013 – Published: 14 February 2013

Correspondence to: S. V. Weijs (steven.weijs@epfl.ch)

Published by Copernicus Publications on behalf of the European Geosciences Union.

Abstract

When inferring models from hydrological data or calibrating hydrological models, we might be interested in the information content of those data to quantify how much can potentially be learned from them. In this work we take a perspective from (algorithmic) information theory (AIT) to discuss some underlying issues regarding this question. In the information-theoretical framework, there is a strong link between information content and data compression. We exploit this by using data compression performance as a time series analysis tool and highlight the analogy to information content, prediction, and learning (understanding is compression). The analysis is performed on time series of a set of catchments, searching for the mechanisms behind compressibility.

We discuss both the deeper foundation from algorithmic information theory, some practical results and the inherent difficulties in answering the question: "How much information is contained in this data?".

The conclusion is that the answer to this question can only be given once the following counter-questions have been answered: (1) Information about which unknown quantities? (2) What is your current state of knowledge/beliefs about those quantities?

Quantifying information content of hydrological data is closely linked to the question of separating aleatoric and epistemic uncertainty and quantifying maximum possible model performance, as addressed in current hydrological literature. The AIT perspective teaches us that it is impossible to answer this question objectively, without specifying prior beliefs. These beliefs are related to the maximum complexity one is willing to accept as a law and what is considered as random.

1 Introduction

How much information is contained in hydrological time series? This question is not often explicitly asked, but is actually underlying many challenges in hydrological modeling and monitoring. Information content of time series is for example relevant for decisions

HESSD
10, 2029–2065, 2013

regarding what to measure and where, to achieve optimal monitoring network designs (Alfonso et al., 2010a,b; Mishra and Coulibaly, 2010; Li et al., 2012). Also in hydrological model inference and calibration, the question can be asked, to decide how much model complexity is warranted by the data (Jakeman and Hornberger, 1993; Vrugt et al., 2002; Schoups et al., 2008; Laio et al., 2010; Beven et al., 2011).

There are, however, some issues in quantifying information content of data. Although the question seems straightforward, the answer is not. This is partly due to the fact that the question is not completely specified. The answers found in data are relative to the question that one asks the data. Moreover, the information content of those answers depends on how much was already known before the answer was received. An objective assessment of information content is therefore only possible when prior knowledge is explicitly specified.

In this paper, we take a perspective from (algorithmic) information theory, (A)IT, on quantifying information content in hydrological data. This puts information content in the context of data compression. The framework naturally shows how specification of the question and prior knowledge enter the problem and to what degree an objective assessment is possible using tools from information theory. The illustrative link between information content and data compression is elaborated in practical explorations of information content, using common compression algorithms.

This paper must be seen as a first exploration of the compression framework to define information content in hydrological time series, with the objective of introducing the analogies and showing how they work in practice. The results will also serve as a benchmark in a follow-up study Weijs et al. (2013), where new compression algorithms are developed that employ hydrological knowledge to improve compression.

2 Information content, patterns, and compression of data

From the framework of information theory, originated by Shannon (1948), we know that information content of a message, data point or observation, can be equated to

HESSD
10, 2029–2065, 2013

suprisal, defined as $-\log P$, where P is the probability assigned to the event before observing it. To not lengthen this paper too much, we refer the reader to Shannon (1948); Cover and Thomas (2006) for more background on information theory. See also Weijs et al. (2010a,b) for introduction and interpretations of information measures in the context of hydrological prediction and model calibration. We also refer the reader to Singh and Rajagopal (1987); Singh (1997); Ruddell et al. (2013), for more references on applications of information theory in the geosciences. In the following, the interpretation of information content as description length is elaborated.

2.1 Information theory: entropy and code length

Data compression seeks to represent the most likely events (most frequent characters in a file) with the shortest codes, yielding the shortest total code length. As is the case with dividing high probabilities, also short codes are a limited resource that has to be allocated as efficiently as possible. When required to be uniquely decodable, short codes come at the cost of longer codes elsewhere. This follows from the fact that such codes must be prefix free, i.e. no code can be the first part (prefix) of another one. This is formalized by the following theorem of McMillan (1956), who generalized the inequality (Eq. 1) of Kraft (1949) to all uniquely decodable codes.

$$\sum_i A^{-l_i} \leq 1 \tag{1}$$

in which A is the alphabet size (2 in the binary case) and l_i is the length of the code assigned to event i. In other words, one can see the analogy between prediction and data compression through the similarity between the scarcity of short codes and the scarcity of large predictive probabilities. Just as there are only 4 probabilities of $\frac{1}{4}$ available, there are only 4 prefix-free binary codes as short as $-\log_2 \frac{1}{4} = 2$ (see Fig. 1, code A). In contrast to probabilities, which can be chosen freely, the code lengths are limited to integers. For example, code B in the table uses one code of length 1, one of length

HESSD
10, 2029–2065, 2013

2 and two of length 3, we can verify that it sharply satisfies Eq. (1), using $A = 2$, we find $1 \cdot 2^{-1} + 1 \cdot 2^{-2} + 2 \cdot 2^{-3} = 1 \leq 1$

In Fig. 1, it is shown how the total code length can be reduced, assigning codes of varying length depending on occurrence frequency. As shown by Shannon (1948), if every value could be represented with one code, allowing for non-integer code lengths, the optimal code length for an event i is $l_i = \log\left(1/p_i\right)$. The minimum average code length is the expectation of this code length over all events, H bits per sample, where H can be recognized as the entropy of the distribution (Shannon, 1948; Cover and Thomas, 2006), which is a lower bound for the average description length.

$$H(\boldsymbol{p}) = E_p\{l\} = \sum_{i=1}^{n} p_i \log \frac{1}{p_i} \tag{2}$$

However, because in practice the code lengths often have to be rounded to an integer number of bits, some overhead will occur. The rounded coding would be optimal for a probability distribution of events

$$q_i = \frac{1}{2^{l_i}} \forall i, \tag{3}$$

such as frequencies II in Fig. 1. In this equation, q_i is the i^{th} element of the probability mass function $\boldsymbol{q}$ for which the code would be optimal and l_i is the code length assigned to event i. The overhead in the case where $\boldsymbol{p} \neq \boldsymbol{q}$ is $D_{\mathrm{KL}}(\boldsymbol{p}\|\boldsymbol{q})$, yielding a total average code length of

$$H(\boldsymbol{p}) + D_{\mathrm{KL}}(\boldsymbol{p}\|\boldsymbol{q}) \tag{4}$$

bits per sample. In general, if a wrong probability estimate is used, the number of bits per sample is increased by the Kullback-Leibler divergence from the true to the estimated probability mass function. This is extra description length is analogous to the reliability term in the decomposition of an information-theoretical score for forecast

HESSD
10, 2029–2065, 2013

quality presented in Weijs et al. (2010b); see Appendix A for an elaboration of this connection.

For probability distributions that do not coincide with integer ideal code lengths, the algorithm known as Huffman coding (Huffman, 1952) was proven to be optimal for value by value compression. It finds codes of an expected average length closest to the entropy-bound and is applied in popular compressed picture and music formats like jpg, tiff, mp3 and wma. For a good explanation of the workings of this algorithm, the reader is referred to Cover and Thomas (2006). In Fig. 1, code A is optimal for probability distribution I and code B is optimal for the distribution II. Both these codes achieve the entropy bound. Code B is also an optimal Huffman code for the distribution III (last column in Fig. 1). Although the expected code length is now more than the entropy, it is impossible to find a shorter code. The overhead is equal to the Kullback-Leibler divergence from the true distribution (III) to the distribution for which the code would be optimal.

$$D_{KL}(\mathrm{III}\|\mathrm{II}) = D_{KL}((0.4, 0.05, 0.35, 0.2)\|(0.5, 0.25, 0.125, 0.125)) = 0.4106$$

If the requirement that the codes are value by value (one code for each observation) is relaxed, blocks of values can be grouped together to approach an ideal probability distribution. When the series are long enough, entropy coding methods like Shannon and Huffman coding using blocks can get arbitrarily close to the entropy bound (Cover and Thomas, 2006). This happens for example in arithmetic coding, where the entire time series is coded as one single number.

2.2 Dependency

If the values in a time series are not independent, however, the dependencies can be used to achieve even better compression. This high compression results from the fact that for dependent values, the joint entropy is lower than the sum of entropies of individual values. In other words, average uncertainty per value decreases, when all

HESSD
10, 2029–2065, 2013

the other values in the series are known, because we can recognize patterns in the series, that therefore contain information about themselves. Hydrological time series often show strong internal depen encies, leading to better compression and better prediction. Consider, for example, the case where you are asked to assign probabilities (or code lengths) to possible streamflow values on 12 May 1973. In one case, the information offered is the dark-colored climatological histogram (Fig. 2 on the right), in the second case, the time series is available (the left of the same figure). Obviously, the e pected compression and expected return for the bets are better in the second case, which shows the value of exploiting dependencies in the data. The surprise ($-\log P_{\text{true value}}$) upon hearing the true value is 3.72 bits in case the guessed distribution was assumed and 4.96 bits when using the climate as prior. These surprises are equivalent to the divergence scores proposed in Weijs et al. (2010b).

Another example are the omitted characters that the careful reader may (not) have found in the previous paragraph. There are 48 different characters used, but the entropy of the text is 4.3 bits, far less than $\log(48) = 5.6$, because of for example the relatively high frequencies of the space (16 %) and the letter "e" (13 %). Although the entropy is more than 4 bits, the actual uncertainty about the missing letters is far less for most readers, because the structure in the text is similar to English language and that structure can be used to predict the missing characters. On the one hand this means that English language is compressible and therefore fairly inefficient. On the other hand this redundancy leads to more robustness in the communication, because even with many typographical errors, the meaning is still clear. If English were 100 % efficient, any error would obfuscate the meaning.

In general, better prediction, i.e. less surprise, gives better results in compression. In water resources management and hydrology we are generally concerned with predicting one series of values from other series of values, like predicting streamflow (Q) from precipitation (P) and potential evaporation (E_p). In terms of data compression, knowledge of P and E_p would help compressing Q, but would also be needed for decompression. When P, E_p and Q would be compressed together in one file, the gain

HESSD
10, 2029–2065, 2013

compared to compressing the files individually is related to what a hydrological model learns from the relation between these variables (Cilibrasi, 2007). Similarly, we can try to compress hydrological time series to investigate how much information those compressible series really contain for hydrological modeling.

2.3 Algorithmic information theory

Algorithmic information theory (AIT) was founded as a field by the appearance of three independent publications (Solomonoff, 1964; Chaitin, 1966; Kolmogorov, 1968). The theory looks at data through the lens of algorithms that can produce those data. The basic idea is that information content of an object, like a data set, is related to the shortest way to describe it. Although description length generally depends on the language used, AIT uses the construct of a universal computer introduced by Turing (1937), the Universal Turing Machine (UTM), to show that this dependence takes the form of an additive constant, which becomes relatively less important when more data is available. Chaitin (1975) offered some refinements in the definitions of programs and showed a very complete analogy with Shannon's information theory, including e.g. the relations between conditional entropy and conditional program lengths.

Using the thesis that any computable sequence can be computed by a UTM and that program lengths are universal up to an additive constant (the length of the program that tells one UTM how to simulate another), Kolmogorov (1968) gave very intuitive definitions of complexity and randomness; see also (Li and Vitanyi, 2008) for more background. Kolmogorov defined the complexity of a certain string (i.e. data set, series of numbers) as the length of the minimum computer program that can produce that output on a UTM and then halt. Complexity of data is thus related to how complicated it is to describe. If there are clear patterns in the data, then they can be described by a program that is shorter than the data itself. The majority of conceivable strings of data cannot be "compressed" in this way. Data that cannot be described in a shorter way than literally stating those data is defined as random. This is analogous to the fact that a "law" of nature cannot really be called a law if its statement is more elaborate

HESSD
10, 2029–2065, 2013

than the phenomenon that it explains. A problem with Kolmogorov complexity is that it is incomputable, but can only be approached from above. This is related to the unsolvability of the halting problem (Turing, 1937): it is always possible that there exists a shorter program which is still running (possibly in an infinite loop) that might eventually produce the output and then halt. A paradox that would arise if Kolmogorov complexity were computable is the following definition known as the Berry paradox: "The smallest positive integer not definable in under eleven words".

A shortcut approximation to measuring information content and complexity, is to use a language that is sufficiently flexible to describe any sequence, while still exploiting most of commonly found patterns. While this approach cannot discover all patterns, like a Turing complete description language can, it will offer an upper bound estimation, without having the problems of incomputability. Compressed files are such a language, that use a decompression algorithm to recreate the object in its original, less efficient language. The compressed files can also be seen a programs for a computer, which is simulated by the decompression algorithm on another computer. Since the language is not Turing complete, it is less powerful than the original computer. The constant additional description length for some recursive patterns is replaced by one that grows indefinitely with growing amounts of data. As an example, one can think of trying to compress an ultra high resolution image of a fractal generated by a simple program. Although the algorithmic complexity with respect to the Turing complete executable fractal program language is limited by the size of the fractal program executable and its settings, the losslessly compressed output image will continue to grow with increasing resolution.

3 Compression experiment set-up

In this experiment, a number of compression algorithms is applied to different data sets to obtain an indication of the amount of information they contain. Most compression algorithms use entropy-based coding methods such as introduced in the previous

HESSD
10, 2029–2065, 2013

section, often enhanced by methods that try to discover dependencies and patterns in the data, such as autocorrelation and periodicity.

The data compression perspective indicates that formulating a rainfall-runoff model has an analogy with compressing rainfall-runoff data. A short description of the data will contain a good model about it, whose predictive power outperforms the description length of the model. However, not all patterns found in the data should be attributed to the rainfall-runoff process. For example, a series of rainfall values is highly compressible due to the many zeros (a far from uniform distribution), the autocorrelation, and the seasonality. These dependencies are in the rainfall alone and can tell us nothing about the relation between rainfall and runoff. The amount of information that the rainfall contains for the hydrological model is thus less than the number of data points multiplied by the number of bits to store rainfall at the desired precision. This amount is important because it determines the model complexity that is warranted by the data (Schoups et al., 2008). In fact, we are interested in the Kolmogorov complexity of the data, but this is incomputable. A crude practical approximation of the complexity is the file size after compression by some commonly available compression algorithms. This provides an upper bound for the information in the data.

If the data can be regenerated perfectly from the compressed (colloquially referred to as zipped) files, the compression algorithm is said to be lossless. In contrast to this, lossy compression introduces some small errors in the data. Lossy compression is mainly used for various media formats (pictures; video; audio), where these errors are often beyond our perceptive capabilities. This is analogous to a model that generates the observed values to within measurement precision, which could be a way to account for uncertainties in observation (Beven and Westerberg, 2011; Westerberg et al., 2011; Weijs and Van de Giesen, 2011). In this paper, we consider only lossless compression. Roughly speaking, the file size that remains after compression, gives an upper bound for the amount of information in the time series. Actually, also the code-length of the decompression algorithm should be counted towards this file size (cf. a self-extracting archive). In the present exploratory example the inclusion of the algorithmic complexity

HESSD
10, 2029–2065, 2013

of the decompression algorithm is not so relevant since the algorithm is general purpose and not biased towards hydrological data. This means that any specific pattern still needs to be stored in the compressed file. The compression algorithms will be mainly used to explore the difference in information content between different signals.

3.1 Quantization

Due to the limited amount of data, quantization is necessary to make meaningful estimates of the distributions, which are needed to calculate the amount of information and compression. This is analogous to the maximum number of bins permitted to draw a representative histogram. As will be argued in the discussion, different quantizations imply different questions for which the information content of the answers is analyzed. All series were first quantized to 8 bit precision, using a simple linear quantization scheme (Eq. 5). Using this scheme, the series were split into $2^8 = 256$ equal intervals and converted into an 8 bit unsigned integer (an integer ranging from 0 to 255 that can be stored in 8 binary digits).

$$x_{\text{integer}} = \lfloor 0.5 + 255 \frac{x - \min x}{\max x - \min x} \tag{5}$$

These can be converted back to real numbers using

$$x_{\text{quantized}} = \left(\frac{\max x - \min x}{255}\right) x_{\text{integer}} + \min x \tag{6}$$

Because of the limited precision achievable with 8 bits , $x_{\text{quantized}} \neq x$. This leads to rounding errors, which can be quantified as a signal to noise ratio (SNR). The SNR is the ratio of the variance of the original signal to the variance of the rounding errors.

$$\text{SNR} = \frac{\frac{1}{n}\sum_{t=1}^{n}(x_t - \bar{x})^2}{\frac{1}{n}\sum_{t=1}^{n}\left(x_t - x_{t,\text{quantized}}\right)^2} \tag{7}$$

Because the SNR can have a large range, it is usually measured in the form of a logarithm, which is expressed in the unit decibel: $\text{SNR}_{\text{dB}} = 10\log_{10}(\text{SNR})$.

HESSD
10, 2029–2065, 2013

3.2 Compression algorithms

The algorithms that were used are a selection of commonly available compression programs and formats. Below are very short descriptions of the main principles and main features of each of the algorithms used and some references for more detailed descriptions. The descriptions are sufficient to understand the most significant pattern in the results. It is beyond the scope of this paper to describe the algorithms in detail.

- ARJ: Uses LZ77 (see LZMA) with sliding window and Huffman coding.
- WAVPACK: Is a lossless compression algorithm for audio files.
- JPG: The Joint Photography Experts Group created the JPEG standard, which includes a range of lossless and lossy compression techniques. Here the lossless coding is used, which uses a Fourier-like type of transform (Discrete cosine transform) followed by Huffman coding of the errors).
- HDF_RLE: HDF (hierarchical data format) is a data format for scientific data of any form, including pictures, time series and metadata. It can use several compression algorithms, including run length encoding (RLE). RLE replaces sequences of reoccurring data with the value and the number of repetitions. It would therefore be useful to compress pictures with large uniform surfaces and rainfall series with long dry periods.
- PPMD: A variant of Prediction by Partial Matching, implemented in the 7Zip program. It uses a statistical model for predicting each value from the preceding values using a variable sliding window. Subsequently the errors are coded using Huffman Coding.
- LZMA: The Lempel-Ziv-Markov chain algorithm combines the Lempel-Ziv algorithm, LZ77 (Ziv and Lempel, 1977), with a Markov-Chain model. LZ77 uses a sliding window to look for reoccurring sequences, which are coded with references

HESSD
10, 2029–2065, 2013

to the previous location where the sequence occurred. The method is followed by range coding. Range coding (Martin, 1979) is an entropy-coding method which is mathematically equivalent to arithmetic coding (Rissanen and Langdon, 1979), it has less overhead than Huffman coding.

- BZIP2: Uses the Burrows and Wheeler (1994) block sorting algorithm in combination with Huffman-Coding.
- PNG: Portable Network Graphics (PNG) uses a filter based on prediction of one pixel from the preceding pixels. Afterward, the prediction errors are compressed by the algorithm "Deflate" which uses dictionary coding (matching repeating sequences) followed by Huffman coding.
- TIFF: A container image format that can use several compression algorithms. In this case PackBits compression was used, which is a form of run length encoding.

3.3 Experiment A: comparison on generated and hydrological time series

In the first experiment, the algorithms are tested on a real world hydrological data set from Leaf River (MS, USA) consisting of rainfall, potential evaporation and streamflow. See e.g. Vrugt et al. (2003) for a description of this data set. As a reference, various artificially generated series where used. The generated series consist of 50 000 values, while the time series of the Leaf River data set, contains 14 610 values (40 yr of daily values). The following series where used in this experiment. All are quantized directly with the linear scheme using Eq. (5).

3.4 Experiment B: Compression with a hydrological model

The second experiment is a first exploration of jointly compressing time series. In the previous experiment single time series were compressed to obtain an indication of their information content. Given the connection between modeling and data compression, a hydrological model should in principle be able to compress hydrological data. This

HESSD
10, 2029–2065, 2013

can be useful to identify good models in information-theoretical terms, but can also be useful for actual compression of hydrological data. Although a more detailed analysis is left for future work, we perform a first test of estimating the performance of hydrological models using data compression tools.

The hydrological model HYMOD was used to predict discharge from rainfall for the Leaf River data set; see e.g. Vrugt et al. (2009) for a description of model and data. Subsequently, the modeled discharges were quantized using the same quantization scheme as the observed discharges. An error signal was defined by subtracting the modeled (Q_{mod}) from the observed (Q) quantized discharge. This gives a signal that can range from −255 to +255, but because the errors are sufficiently small, ranged from −55 to +128, which allows for 8 bit coding. Because the observed discharge signal (Q) can be reconstructed from the precipitation time series (P), the model, and the stored error signal (Q_{err}), the model could enable compression of the data set consisting of P and Q. In the experiment we test whether the error series is indeed more compressible than the original time series of Q.

3.5 Experiment C: Compression of hydrological time series from the MOPEX data set

In a third experiment, we looked at the spatial distribution of compressibility for daily streamflow and rainfall data in the 431 river basins across the continental USA, as contained in the MOPEX data set. This should give some indication about the information content or complexity of the time series. For these experiments, the streamflow values are log-transformed before quantization, to reflect the heteroscedastic uncertainty in the measurements. Missing values, which were infrequent, were removed from the series. Although this can have some impact on the ability to exploit autocorrelation and periodicity, the effect is deemed to be small and has a smaller influence than other strategies such as replacing the missing values by zero or a specific marker. Results of this compression experiment are presented in Sect. 4.3.

HESSD
10, 2029–2065, 2013

4 Results of the compression experiments

This section shows results from the compression analysis for single time series. Also an example of compression of discharge, using a hydrological model in combination with knowledge of rainfall, is shown.

4.1 Results A: generated data

As expected, the file sizes after quantization are exactly equal to the number of values in the series, as each value is encoded by one byte (8 bits) and stored in binary raw format. From the occurrence frequencies of the 256 unique values, the entropy of their distribution was calculated. Normalized with the maximum entropy of 8 bits, the fractions in row 3 of Table 2 give an indication of the entropy bound for the ratio of compression achievable by value by value entropy encoding schemes such as Huffman coding, which do not use temporal dependencies.

The signal to noise ratios in row 4 give an indication of the amount of data corruption that is caused by the quantization. As a reference, the uncompressed formats BMP (Bitmap), WAV (Waveform audio file format), and HDF (Hierarchical Data Format) are included, indicating that the file size of those formats, relative to the raw data, does not depend on what data are in them, but does depend on the amount of data, because they have a fixed overhead that is relatively smaller for larger files.

The results for the various lossless compression algorithms are shown in rows 7–17. The numbers are the percentage of the file size after compression, relative to the original file size (a lower percentage indicates better compression). The best compression ratios per time series are highlighted. From the result it becomes clear that the constant, linear and periodic signals can be compressed to a large extent. Most algorithms achieve this high compression, although some have more overhead than others. The uniform white noise is theoretically incompressible, and indeed none of the algorithms appears to know a clever way around this. In fact, the smallest file size is achieved by the WAV format, which does not even attempt to compress the data and has a relatively

HESSD
10, 2029–2065, 2013

small file header (meta information about the file format). The Gaussian white noise is also completely random in time, but does not have a uniform distribution. Therefore the theoretical limit for compression is the entropy bound of 86.3 %. The WAVPACK algorithm gets closest to the theoretical limit, but also several file archiving algorithms (ARJ, PPMD, LZMA BZIP2) approach that limit very closely. This is because they all use a form of entropy coding as a back-end (Huffman and Range coding). Note that the compression of this non-uniform white noise signal is equivalent to the difference in uncertainty or information gain due to knowledge of the occurrence frequencies of all values (the climate), compared to a naive uniform probability estimate; cf. the first two bars in Fig. 1 of Weijs et al. (2010a).

The results for the hydrological series firstly show that the streamflow series is better compressible than the precipitation series. This is remarkable, because the rainfall series has the lower entropy. Furthermore it can be seen that for the rainfall series, the entropy-bound is not achieved by any of the algorithms, presumably because of the overhead caused by the occurrence of 0 rainfall more than 50 percent of the time, see Eqs. (3) and (4). Further structure like autocorrelation and seasonality can not be used sufficiently to compensate for this overhead. In contrast to this, the streamflow series can be compressed to well below the entropy bound (27.7 % vs. 42.1 %), because of the strong autocorrelation in the data. These dependencies are best exploited by the PPMD algorithm, which uses a local prediction model that apparently can predict the correlated values quite accurately. Many of the algorithms cross the entropy bound, indicating that they use at least part of the temporal dependencies in the data.

4.2 Results B: Compression with a hydrological model

We analyzed the time series of Q and P for leaf river, along with the modeled Q (Q_{mod}) and its errors (Q_{err}). In Table 3, the entropies of the signals are shown. The second row shows the resulting file size as percentage of the original file size for the best compression algorithm for each series (PPMD or LZMA).

HESSD
10, 2029–2065, 2013

The table also shows the statistics for the series where the order of the values was randomly permuted (Q^{perm} and $Q_{\text{err}}^{\text{perm}}$). As expected this does not change the entropy, because that depends only on the histograms of the series. In contrast, the compressibility of the signals is significantly affected, indicating that the compression algorithms made use of the temporal dependence for the non-permuted signals. The joint distribution of the modeled and observed discharges was also used to calculate the conditional entropy $H(Q|Q_{\text{mod}})$. It must be noted, however, that this conditional entropy is probably underestimated, as it is based on a joint distribution with 255^2 probabilities estimated from 14 610 value pairs. This is the cost of estimating dependency without limiting it to a specific functional form. The estimation of mutual information needs more data than Pearson correlation, because the latter is limited to a linear setting and looks at variance rather than uncertainty. In the description length, the underestimation of $H(Q|Q_{\text{mod}})$ is compensated by the fact that the dependency must be stored by the entire joint distribution. If representative for the dependence in longer data sets, the conditional entropy gives a theoretical limit of compressing Q with knowledge of P and the model, while not making use of temporal dependence.

A somewhat unexpected result is that the errors seem more difficult to compress (31.5 %) than the observed discharge itself (27.7 %), even though the entropy is lower. Apparently the reduced temporal dependence in the errors (lag-1 autocorrelation coefficient $\rho = 0.60$), compared to that of the discharge ($\rho = 0.89$), offsets the gain in compression due to the lower entropy of the errors. Possibly, the temporal dependence in the errors becomes to complex to be detected by the compression algorithms. Further research is needed to determine the exact cause of this result, which should be consistent with the theoretical idea that the information in P should reduce uncertainty in Q. The Nash-Sutcliffe Efficiency (NSE) of the model over the mean is 0.82, while the NSE over the persistence forecast ($Q_{\text{mod}}(t) = Q_{t-1}$) is 0.18 (see Schaefli and Gupta, 2007), indicating a reasonable model performance. Furthermore, the difference between the conditional entropy and the entropy of the errors could indicate that an additive error model is not the most efficient way of coding and consequently not the most efficient

HESSD
10, 2029–2065, 2013

tool for probabilistic prediction. The use of for example heteroscedastic probabilistic forecasting models (e.g. Pianosi and Soncini-Sessa, 2009) for compression is left for future work.

4.3 Results C: MOPEX data set

For the time series of the quantized scaled log streamflow and scaled quantized rainfall of the MOPEX basins, from now on simply referred to as streamflow (Q) and rainfall (P), for brevity, the compressibility and entropy show clear spatial patterns. For most of the streamflow time series, the entropy is close to 8 bits, indicating that the frequency distribution of the preprocessed streamflow does not diverge much from a uniform distribution. An exception are the basins in the central part of the USA, which show lower entropy time series due to high peaks and relatively long, low base flow periods. Also for the rainfall, entropy values are lower in this region due to longer dry spells; see Fig. 3.

Compression beyond the entropy bound can be achieved by using temporal patterns. This is visible in Fig. 4, where the compression ratio of the best performing algorithm is visualized relative to the entropy of the signals. Different algorithms are specialized in describing different kinds of patterns, so the map of best performing algorithms (Fig. 5) can be used as an indication for which types of patterns are found in data. In Fig. 6, some of two influences on compression rate are shown. Firstly, due to temporal dependencies in the streamflow, the conditional entropy given the previous value $H(Q_t|Q_{t-1})$, known as the entropy rate $H'(Q)$, is much lower than the entropy itself. This could theoretically lead to a compression describing the signal with $H'(Q)$ bits per time step. However, because of the relatively short length of the time series compared to the complexity of the model that describes it (a two dimensional 256 bin histogram), this compression is not reached in practice, because the model needs to be stored too. This is a natural way of accounting for model complexity in the context of estimating information content of data.

HESSD
10, 2029–2065, 2013

5 Discussion

The data compression results give an indication of the information content or complexity of the data. Eventually, these may be linked to climate and basin characteristics and become a tool for hydrological time series analysis and inference. Although information theory may eventually provide a solid foundation for hydrological modeling, it is also important to first consider the limitations such approaches. In this paper, we discuss some inherent issues in quantifying the information content, which makes the results subjective and not straightforward to analyze.

5.1 How much information is contained in this data?

From the presented theoretical background, results, and analysis it can be concluded that although information theory can quantify information content, the outcome depends on a number of subjective choices. These subjective choices include the quantization, auxiliary data, and prior knowledge used.

The quantization can be linked to what question the requested information answers. When quantizing streamflow into 256 equally sized classes, the question that is implicitly posed is: "in which of these equally spaced intervals does the streamflow fall?". When the logarithm of the streamflow is used instead, the intervals change, and therefore also the questions change. The question requests more absolute precision on the lower flows than on the higher flows. The information contained in the answers given by the data, i.e. the information content of the time series, depends on the question that is asked.

The information content of time series depends also what prior knowledge one has about the answers to the question asked. If one knows the frequency distribution but has no knowledge of surrounding values, the prior knowledge takes the form of a probability distribution that matches the observed frequencies. In that case, the expected information content of each observation is given by the entropy of the frequency

HESSD
10, 2029–2065, 2013

distribution. The entropy in bits gives the limit of the minimum average space per observation needed to store a long i.i.d. time series of that distribution.

In many situations in practice, however, prior knowledge does not include knowledge of the occurrence frequencies, or does include more knowledge than frequencies alone, e.g. temporal dependencies. In the first case the information content of the data should also include the knowledge gained from observing the frequencies. Also in compression, optimal coding table, which depends on the frequencies, should be stored and adds to the file size. One could see the histogram as a simple form of a model that is inferred from the data. The model generally forms part of the information content.

In the second case, temporal dependencies reduce the average information content per observation. Also when the form of the temporal dependencies are not know a priori, but inferred from the data, they can decrease the information content, if the gain in compression offsets the space needed to store the model describing the dependencies. In the theoretical framework of algorithmic information theory, model and data are unified in one algorithm (one could see as a self-extracting archive) and the length of the shortest algorithm that reproduces the data is the information content, or Kolmogorov Complexity (Kolmogorov, 1968).

Flexible data compression algorithms, such as used in this paper, are able to give an upper bound for the information content of hydrological data, because they are not specifically tuned towards hydrological data. All patterns inferred from the data are stored in the compressed file and very little is considered as prior information. Theoretically, prior information can be explicitly fed to new compression algorithms in the form of auxiliary data files (e.g. rainfall to compress runoff) or function libraries (e.g. hydrological models), which should reduce information content of the data due to the increase in prior knowledge.

Summarizing, we can state that information content of data depends on (1) what question we ask the data, and (2) how much is already known about the answer before seeing the data.

HESSD
10, 2029–2065, 2013

5.2 Aleatoric and epistemic uncertainty

In current hydrological literature, attempts are sometimes made to separate epistemic (due to incomplete knowledge of the process) from aleatoric (the "inherent" randomness in the system) uncertainty. The approach to answer this question is equivalent to trying to separate pattern from scatter (signal from noise) in high dimensional data spaces, to see how much of the variability can potentially be explained by any model.

However, the inherent problem in answering this question is the subjectivity of what we call pattern and what we call scatter. Although model complexity control methods can give guidelines on how much pattern can be reasonably inferred from data, they usually do not account for prior knowledge. This prior knowledge may affect to a large degree what is considered a pattern, for example by constraining the model class that is used to search for patterns or by introducing knowledge of underlying physics. In the algorithmic information theory sense, this can be equivalently expressed either as prior knowledge favoring certain long (so otherwise unlikely) programs that describe the data, or prior knowledge favoring a certain reference computer or language, which offers a shorter description for that specific pattern.

As a somewhat extreme, unlikely, but illustrative example, consider that we encounter 100 consecutive digits of π as a streamflow time series. Our prior hydrological knowledge would indicate those values as random, and containing a large amount of information (no internal dependence or predictability). With different prior knowledge, however, for example that the data is the output of a computer program authored by a student, we would consider the data as having a pattern, and could use this to make predictions or compress the data (by inferring one of the possible programs the enumerate digits of π as a probable source of the data). There would be little surprise in the second half of the data, given the first.

HESSD
10, 2029–2065, 2013

6 Conclusions

Determining information content of data is a similar process as building a model of the data or compressing the data. These processes are subject to prior knowledge and therefore this knowledge should be explicitly considered in determining information content. Quantization of the data can be seen as a formulation of the question the data is asked to give information about. Upper bounds for information content for that question can than be found using compression algorithms on the quantized data.

A hydrological model actually is such a compression tool. It makes use of the dependencies between for example rainfall and streamflow. The patterns that are already present in the rainfall reduce the information that the hydrological model can learn from: a long dry period could for example be summarized by one parameter for an exponential recession curve in the streamflow. The information available for a rainfall runoff model could theoretically be estimated by comparing the file size of compressed rainfall plus the file size of compressed streamflow with the size of a file where rainfall and streamflow are compressed together, exploiting their mutual dependencies. We could denote this as:

$$\text{learnable info} = |\text{ZIP}(P)| + |\text{ZIP}(Q)| - |\text{ZIP}(P,Q)| \quad (8)$$

where $|\text{ZIP}(X)|$ stands for the file size of a theoretically optimal compression of data X, which includes the size of the decompression algorithm. This brings us back to the ideas of algorithmic information theory, which uses program lengths that reproduce data on computers (Turing machines). The shortening in description length when merging input and output data, i.e. the compression progress, could be seen as the amount of information learned by modeling. The hydrological model that is part of the decompression algorithm embodies the knowledge gained from the data.

Further explorations of these ideas from algorithmic information theory are expected to put often-discussed issues in hydrological model inference in a wider perspective with more general and robust foundations.

HESSD
10, 2029–2065, 2013

Appendix A

Correspondence of resolution – reliability – uncertainty decomposition to compression and structure

In this appendix, we give a data-compression interpretation of Kullback-Leibler divergence as a forecast skill score and its decomposition into uncertainty, reliability and resolution, as proposed in Weijs et al. (2010b). As noted in Sect. 2.1, when observations have distribution $\boldsymbol{p}$, but an optimal fixed coding is chosen assuming the distribution is $\boldsymbol{q}$, the expected average code length per observation is given by

$$H(\boldsymbol{p}) + D_{\mathrm{KL}}(\boldsymbol{p}\|\boldsymbol{q}).$$

The code length is related to the remaining uncertainty, i.e. the missing information, i.e. the amount of information that remains to be specified to reproduce the data. In terms of forecast evaluation and the decomposition presented in Weijs et al. (2010b), using the same notation, this remaining uncertainty is the divergence score associated with a forecast with zero resolution (forecasts do not change), and non-zero reliability (forecast distribution $\boldsymbol{f}$ is not equal to climatological distribution $\bar{\boldsymbol{o}}$)

$$\mathrm{DS} = H(\bar{\boldsymbol{o}}) + D_{\mathrm{KL}}(\bar{\boldsymbol{o}}\|\boldsymbol{f}) = \mathrm{UNC} + \mathrm{REL}.$$

The resolution term, given by the Kullback-Leibler divergence from the marginal distribution $\bar{\boldsymbol{o}}$ to the conditional distributions of observations $\bar{\boldsymbol{o}}_k$, given forecast $\boldsymbol{f}_k$,

$$\mathrm{RES} = D_{\mathrm{KL}}(\bar{\boldsymbol{o}}_k\|\bar{\boldsymbol{o}}),$$

is zero since $\bar{\boldsymbol{o}}_k = \bar{\boldsymbol{o}}$ for an unconditioned, constant forecast (code for compression).

When data with temporal dependencies is compressed, a lower average code length per observation can be achieved, since we can use a dynamically changing coding for next observations, depending on the previous. In terms of forecast quality, this means

HESSD
10, 2029–2065, 2013

that the individual probability estimates now have non-zero resolution. This resolution, which is equivalent to the mutual information between the forecast based on the past time series and the value to code, will reduce the average code length per observation. Since also the individual forecasts will not be completely reliable, the average code length per observation will now have a contribution from each term in the decomposition of the divergence score

$$\mathrm{DS} = H(\bar{\boldsymbol{o}}) + \sum_{k=1}^{K} \frac{n_k}{N} \left[D_{\mathrm{KL}}(\bar{\boldsymbol{o}} \| \boldsymbol{f}_k) - D_{\mathrm{KL}}(\bar{\boldsymbol{o}}_k \| \bar{\boldsymbol{o}}) \right] = \mathrm{UNC} + \mathrm{REL} - \mathrm{RES}$$

where n_k is the number of observations for which unique forecast no. k is given and N is the total number of observations. When compressing data, however, the prediction model that describes the temporal dependence needs to be stored as well. Therefore, the average total code length per data-point will become

$$\begin{aligned} \mathrm{DS} &= H(\bar{\boldsymbol{o}}) + \sum_{k=1}^{K} \frac{n_k}{N} \left[D_{\mathrm{KL}}(\bar{\boldsymbol{o}} \| \boldsymbol{f}_k) - D_{\mathrm{KL}}(\bar{\boldsymbol{o}}_k \| \bar{\boldsymbol{o}}) \right] + L(\mathrm{model})/N \\ &= \mathrm{UNC} + \mathrm{REL} - \mathrm{RES} + \mathrm{COMPLEXITY}/N \end{aligned}$$

where L (model) is the length of the model algorithm. Although this model length is language dependent, it is known from AIT that this dependence is just an additive constant, and can be interpreted as the prior knowledge encoded in the language. If the language is not specifically geared towards a certain type of data, the total code length will give a fairly objective estimate of the amount of new information in the data, which cannot be explained from the data itself. The number of bits per sample needed to store data can therefore be interpreted as a complexity-penalized version of the divergence score presented in Weijs et al. (2010a,b), applied to a predictions of the data based on previous time steps. We can make the following observations. Firstly, data can only be compressed if there is a pattern, i.e. something that can be described be an algorithm where the resolution or gain in description efficiency or predictive power

HESSD
10, 2029–2065, 2013

outweighs the loss due to complexity. Secondly, the data compression view naturally leads to the notion that we have to penalize model complexity when evaluating the predictive performance of models.

Acknowledgements. Steven Weijs is a beneficiary of a postdoctoral fellowship from the AXA research fund, which is gratefully acknowledged. Funding from the Swiss Science Foundation, the NCCR-MICS and CCES are also gratefully acknowledged.

References

Alfonso, L., Lobbrecht, A., and Price, R.: Information theory–based approach for location of monitoring water level gauges in polders, Water Resour. Res., 46, W03528, doi:10.1029/2009WR008101, 2010a. 2031

Alfonso, L., Lobbrecht, A., and Price, R.: Optimization of water level monitoring network in polder systems using information theory, Water Resour. Res., 46, W12553, doi:10.1029/2009WR008953, 2010b. 2031

Beven, K. and Westerberg, I.: On red herrings and real herrings: disinformation and information in hydrological inference, Hydrol. Process., 25, 1676–1680, doi:10.1002/hyp.7963, 2011. 2038

Beven, K., Smith, P. J., and Wood, A.: On the colour and spin of epistemic error (and what we might do about it), Hydrol. Earth Syst. Sci., 15, 3123–3133, doi:10.5194/hess-15-3123-2011, 2011. 2031

Burrows, M. and Wheeler, D. J.: A block-sorting lossless data compression algorithm, Tech. rep., Systems Research Center, Palo Alto, CA, 1994. 2041

Chaitin, G. J.: On the length of programs for computing finite binary sequences, J. ACM, 13, 547–569, 1966. 2036

Chaitin, G. J.: A theory of program size formally identical to information theory, J. ACM, 22, 329–340, 1975. 2036

Cilibrasi, R.: Statistical inference through data compression, Ph.D. thesis, UvA, Amsterdam, 2007. 2036

Cover, T. M. and Thomas, J. A.: Elements of information theory, Wiley-Interscience, New York, 2006. 2032, 2033, 2034

HESSD

10, 2029–2065, 2013

Huffman, D. A.: A Method for the Construction of Minimum-Redundancy Codes, P. IRE, 40, 1098–1101, 1952. 2034
Jakeman, A. J. and Hornberger, G. M.: How much complexity is warranted in a rainfall-runoff model?, Water Resour. Res., 29, 2637–2649, 1993. 2031
Kolmogorov, A. N.: Three approaches to the quantitative definition of information, Int. J. Comput. Math., 2, 157–168, 1968. 2036, 2048
Kraft, L. G.: A device for quantizing, grouping, and coding amplitude-modulated pulses, Master's thesis, Massachusetts Institute of Technology, Dept. of Electrical Engineering, 1949. 2032
Laio, F., Allamano, P., and Claps, P.: Exploiting the information content of hydrological "outliers" for goodness-of-fit testing, Hydrol. Earth Syst. Sci., 14, 1909–1917, doi:10.5194/hess-14-1909-2010, 2010. 2031
Li, C., Singh, V., and Mishra, A.: Entropy theory-based criterion for hydrometric network evaluation and design: Maximum information minimum redundancy, Water Resour. Res., 48, W05521, doi:10.1029/2011WR011251, 2012. 2031
Li, M. and Vitanyi, P. M. B.: An introduction to Kolmogorov complexity and its applications, Springer-Verlag New York Inc, 2008. 2036
Martin, G. N. N.: Range encoding: an algorithm for removing redundancy from a digitised message, in: Video & Data Recording conference, 1979. 2041
McMillan, B.: Two inequalities implied by unique decipherability, IEEE T. Inform. Theory, 2, 115–116, 1956. 2032
Mishra, A. and Coulibaly, P.: Hydrometric network evaluation for Canadian watersheds, J. Hydrol., 380, 420–437, 2010. 2031
Pianosi, F. and Soncini-Sessa, R.: Real-time management of a multipurpose water reservoir with a heteroscedastic inflow model, Water Resour. Res., 45, W10430, doi:10.1029/2008WR007335, 2009. 2046
Rissanen, J. and Langdon, G. G.: Arithmetic coding, IBM J. Res. Dev., 23, 149–162, 1979. 2041
Ruddell, B. L., Brunsell, N. A., and Stoy, P.: Applying Information Theory in the Geosciences to Quantify Process Uncertainty, Feedback, Scale, Eos T. Am. Geophys. Un., 94, p. 56, 2013. 2032
Schaefli, B. and Gupta, H. V.: Do Nash values have value?, Hydrol. Process., 21, 2075–2080, 2007. 2045

HESSD
10, 2029–2065, 2013

Schoups, G., van de Giesen, N. C., and Savenije, H. H. G.: Model complexity control for hydrologic prediction, Water Resour. Res., 44, W00B03, doi:10.1029/2008WR006836, 2008. 2031, 2038
Shannon, C. E.: A mathematical theory of communication, Bell. Syst. Tech. J., 27, 379–423, 1948. 2031, 2032, 2033
Singh, V. P.: The use of entropy in hydrology and water resources, Hydrol. Process., 11, 587–626, 1997. 2032
Singh, V. P. and Rajagopal, A. K.: Some recent advances in application of the principle of maximum entropy (POME) in hydrology, IAHS-AISH P., 194, 353–364, 1987. 2032
Solomonoff, R. J.: A formal theory of inductive inference. Part I, Inform. Control, 7, 1–22, 1964. 2036
Turing, A. M.: On computable numbers, with an application to the Entscheidungsproblem, P. Lond. Math. Soc., 2, 230–265, 1937. 2036, 2037
Vrugt, J. A., Bouten, W., Gupta, H., and Sorooshian, S.: Toward improved identifiability of hydrologic model parameters: The information content of experimental data, Water Resour. Res, 38, 1312, doi:10.1029/2001WR001118, 2002. 2031
Vrugt, J. A., Gupta, H. V., Bouten, W., and Sorooshian, S.: A Shuffled Complex Evolution Metropolis algorithm for optimization and uncertainty assessment of hydrologic model parameters, Water Resour. Res., 39, 1201, doi:10.1029/2002WR001642, 2003. 2041
Vrugt, J. A., Ter Braak, C. J. F., Gupta, H. V., and Robinson, B. A.: Equifinality of formal (DREAM) and informal (GLUE) Bayesian approaches in hydrologic modeling?, Stoch. Env. Res. Risk A., 23, 1011–1026, 2009. 2042
Weijs, S. V. and Van de Giesen, N.: Accounting for observational uncertainty in forecast verification: an information-theoretical view on forecasts, observations and truth, Month. Weather Rev., 139, 2156–2162, doi:10.1175/2011MWR3573.1, 2011. 2038
Weijs, S. V., Schoups, G., and van de Giesen, N.: Why hydrological predictions should be evaluated using information theory, Hydrol. Earth Syst. Sci., 14, 2545–2558, doi:10.5194/hess-14-2545-2010, 2010a. 2032, 2044, 2052
Weijs, S. V., Van Nooijen, R., and Van de Giesen, N.: Kullback–Leibler divergence as a forecast skill score with classic reliability-resolution-uncertainty decomposition, Monthly Weather Review, 138, 3387–3399, 2010b. 2032, 2034, 2035, 2051, 2052
Weijs, S. V., Van de Giesen, N., and Parlange, M. B.: HydroZIP: how hydrological knowledge can be used to improve compression of hydrological data, Entropy, in review, 2013. 2031

HESSD
10, 2029–2065, 2013

Westerberg, I., Guerrero, J., Seibert, J., Beven, K., and Halldin, S.: Stage-discharge uncertainty derived with a non-stationary rating curve in the Choluteca River, Honduras, Hydrol. Process., 25, 603–613, doi:10.1002/hyp.7848, 2011. 2038

Ziv, J. and Lempel, A.: A universal algorithm for sequential data compression, IEEE T. Inform. Theory, 23, 337–343, 1977. 2040

HESSD

10, 2029–2065, 2013

Table 1. Signals used in experiment A.

Signal	Description
constant	contains only 1 value repeatedly
linear	contains a slowly linearly increasing trend
uniform white	is the output from the Matlab® function "rand": uniform white noise
Gaussian white	is the output from the Matlab® function "randn", normally distributed white noise
sine 1	single sinusoidal wave with a wavelength spanning all 50 000 values
sine 100	100 sinusoidal waves with a wavelength spanning 1/100 of 50 000 values
Leaf *P*	daily rainfall series from the catchment of Leaf river (1948–1988)
Leaf *Q*	corresponding daily series of observed streamflow in Leaf river

HESSD
10, 2029–2065, 2013

Table 2. The performance, as percentage of the original file size, of well known compression algorithms on various time series. The best results per signal are highlighted.

Data set	Constant	Linear	Uniform white	Gaussian white	Sine 1	Sin 100	Leaf Q	Leaf P
file size	50 000	50 000	50 000	50 000	50 000	50 000	14 610	14 610
$\frac{H}{\log N}$	0.0	99.9	99.9	86.3	96.0	92.7	42.1	31.0
SNR	NaN	255.0	255.6	108.0	307.4	317.8	42.6	39.9
Uncompressed formats								
BMP	102.2	102.2	102.2	102.2	102.2	102.2	407.4	407.4
WAV	100.1	100.1	100.1	100.1	100.1	100.1	100.3	100.3
HDF_NONE	100.7	100.7	100.7	100.7	100.7	100.7	102.3	102.3
Lossless compression algorithms								
JPG_LS	12.6	12.8	110.6	94.7	12.9	33.3	33.7	49.9
HDF_RLE	2.3	2.7	101.5	101.5	3.2	92.3	202.3	202.3
WAVPACK	0.2	1.9	103.0	87.5	2.9	25.6	38.0	66.2
ARJ	0.3	1.0	100.3	88.0	3.1	1.9	33.7	40.0
PPMD	0.3	2.1	102.4	89.7	3.6	1.4	27.7	36.4
LZMA	0.4	0.9	101.6	88.1	1.9	1.2	31.0	37.8
BZIP2	0.3	1.8	100.7	90.7	3.0	2.3	29.8	40.5
PNG	0.3	0.8	100.4	93.5	1.5	0.8	40.2	50.0
GIF	2.3	15.7	138.9	124.5	17.3	32.0	38.8	45.9
TIFF	2.0	2.4	101.2	101.2	2.9	91.2	201.5	201.5

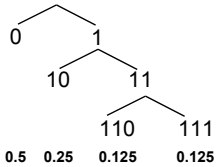
HESSD
10, 2029–2065, 2013

Table 3. Information-theoretical and variance statistics and compression results (remaining file size %) for rainfall-runoff modeling.

statistic	P	Q	Q_{mod}	Q_{err}	$Q\|Q_{mod}$	Q^{perm}	Q_{err}^{perm}
entropy (% of 8 bits)	31.0	42.1	44.9	38.9	26.4	42.1	38.9
best compression (%)	36.4	27.7	25.8	31.5	N.A.	45.4	44.1
std. dev. (range = 256)	11.7	11.6	10.4	4.95	N.A.	11.6	4.95
Autocorrelation ρ	0.15	0.89	0.95	0.60	N.A.	< 0.01	< 0.01

HESSD
10, 2029–2065, 2013

	occurrence frequencies			codes		expected code lengths per value					
event	I	II	III	A	B	A_I	B_I	A_II	B_II	A_III	B_III
CC	0.25	0.5	0.4	00	0	0.5	0.25	1	0.5	0.8	0.4
YY	0.25	0.25	0.05	01	10	0.5	0.5	0.5	0.5	0.1	0.1
GG	0.25	0.125	0.35	10	110	0.5	0.75	0.25	0.375	0.7	1.05
RR	0.25	0.125	0.2	11	111	0.5	0.75	0.25	0.375	0.4	0.6
total	H=2	H=1.75	H=1.74			2	2.25	2	1.75	2	2.15



YYCCCCRRCCGGYYCC

0100001000110100 **CODE A: 16 bits, 2/color**

1000111011010 **CODE B: 14 bits, 1.75/color**

Fig. 1. Assigning code lengths proportional to minus the log of their probabilities leads to optimal compression. Code B is optimal for distribution II, but not for the other distributions. Distribution III has no optimal code that achieves the entropy bound.

HESSD
10, 2029–2065, 2013

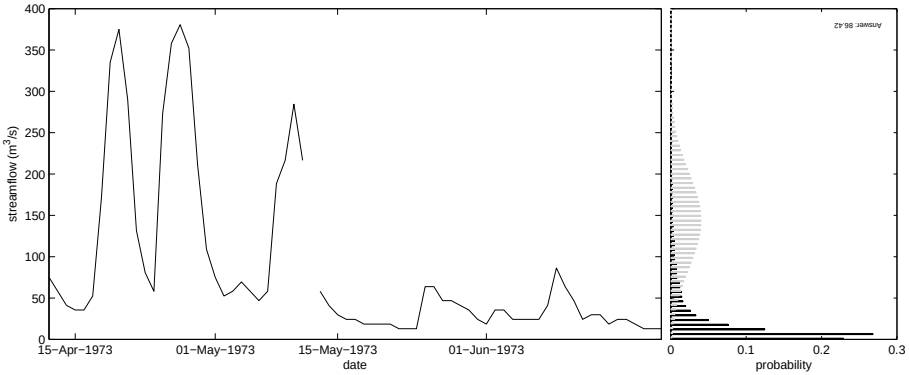


Fig. 2. The missing value in the flow time series can be guessed from the surrounding values (a guess would for example be the grey histogram). This will usually lead to a better guess than one purely based on the occurrence frequencies over the whole 40 yr data set (dark histogram) alone. The missing value therefore contains less information than when assumed independent.

HESSD
10, 2029–2065, 2013

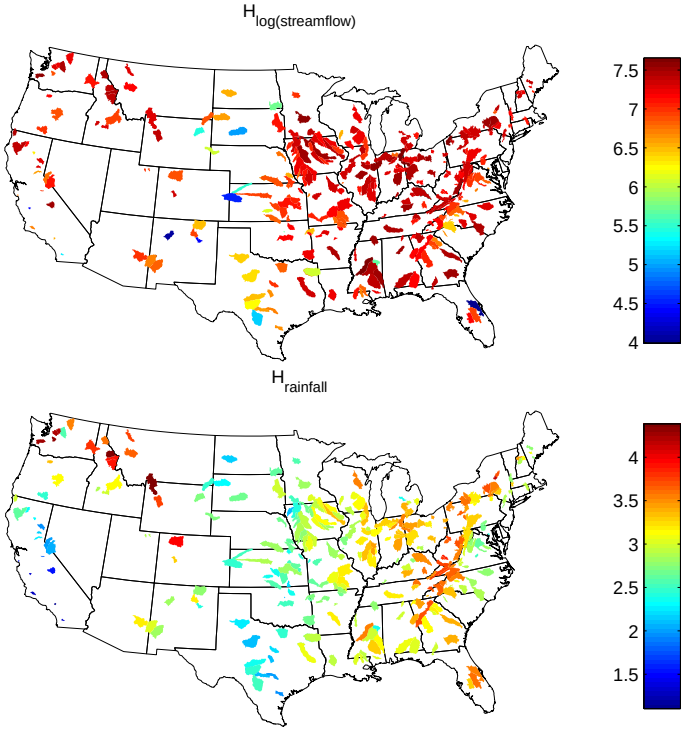


Fig. 3. Spatial distribution of entropy for quantized streamflow and rainfall shows the drier climate in the central part of the USA.

HESSD
10, 2029–2065, 2013

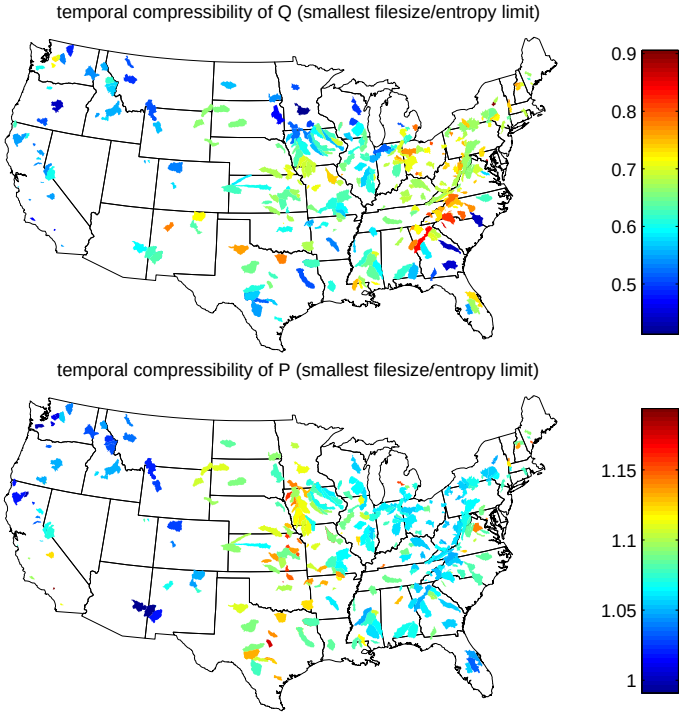


Fig. 4. Spatial distribution the compression size normalized by entropy for streamflow and rainfall, this gives an indication of the amount of temporal structure found in the different basins. The streamflow is better compressible due the strong autocorrelation structure.

HESSD
10, 2029–2065, 2013

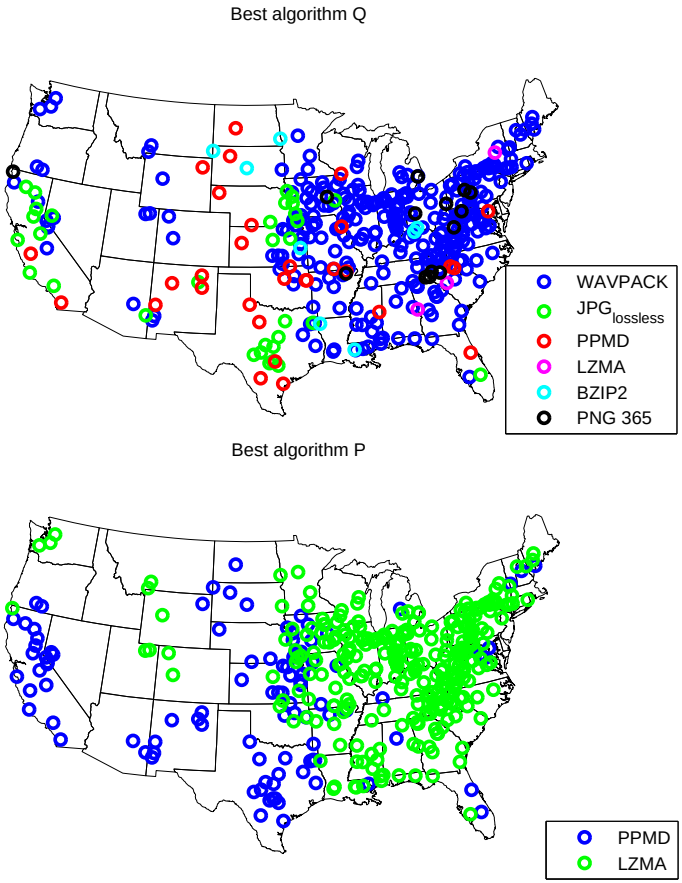


Fig. 5. Spatial distribution of the best performing algorithms for streamflow and rainfall. This can give an indication what type of structure is found in the data. Especially for rainfall, the best performing algorithm is linked to the number of dry days per year. See also Fig. 6.

HESSD
10, 2029–2065, 2013

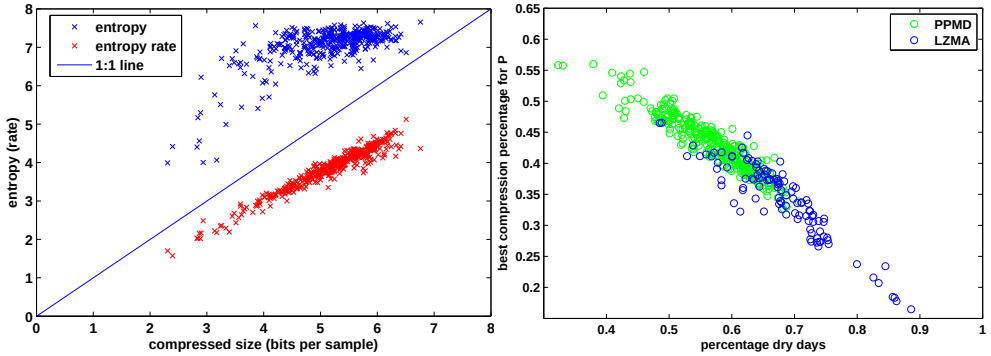


Fig. 6. Left: best compression of Q against entropy and against entropy rate. Temporal dependencies cause better compression than the entropy, but model complexity prevents achieving the entropy rate. Right: the best achieved compression of P depends strongly on the percentage of dry days, mostly through the entropy. Also the best performing algorithm changes with the climate.

HESSD
10, 2029–2065, 2013