Fountain Codes for GNSS

I. Fernández-Hernández; European Commission; Belgium
D. Calle, S. Cancela, A. Fernández, R. Martínez; GMV; Spain
G. Seco-Granados; University Autónoma of Barcelona; Spain
P. Walker; CGI; UK

BIOGRAPHIES

Ignacio Fernández Hernández is the Galileo service definition coordinator at the European Commission, DG ENTR. He is an ICAI engineer, holds an MBA by LBS and a PhD in electronic systems by Aalborg University.

David Calle holds a MSc. in Computer Engineering from the University of Salamanca. He joined GMV in 2008 and he has been working in the GNSS business unit involved in the design and development of GNSS algorithms, applications and systems. He is currently Head of GNSS Services Section coordinating the activities related to the Galileo Commercial Service, Open Service Authentication and High Accuracy provision services.

Simón Cancela holds an MSc in Advanced Mathematics by the Universidad Complutense de Madrid. He joined GMV in 2015 and he has been working in the Galileo Commercial Service Demonstrator validation and experimentation activities and he is currently working on the development of a Commercial Service enhanced PVT resilient platform.

Alba Fernández holds a Degree in Telecommunications Technology and Service Engineering from Universidad Politécnica de Madrid. She joined GMV in 2017 and has participated in activities related with the Galileo Commercial Service. She is currently working in the GNSS Service Centre evolutions.

Rafael Martínez holds a MSc. in Aeronautical Engineering from the University of Seville. He joined GMV in 2016 and he has been working within the Galileo Commercial Service Demonstrator in validation, integration and experimentation activities.

Gonzalo Seco Granados is associate professor with the Dept of Telecom. Eng. of Univ. Autónoma de Barcelona (UAB) and head of the Signal Processing for Navigation and Communications (SPCOMNAV) group. Previoulsy, he was staff member at the Radionavigation Section in ESTEC/ESA, and involved in the Galileo project and in the development of GNSS receivers and applications.

Paul Walker is the Solution Architect at CGI responsible for navigation authentication design and development projects. He received a PhD in Physics in 1996 and has been a software engineer in the space sector since 1999.

ABSTRACT

The dissemination of large messages in a fast and robust way through the GNSS signal in space is relevant for new services such as high-accuracy and authentication, as well as for existing services. Large messages cannot be enclosed into a single word or page, hence without an appropriate strategy of dissemination, losing one portion of the message can imply losing the full message till it is rebroadcast again. Fountain codes are ideal for erasure channels, providing high-robustness against chunk losses. They allow encoding a message into small packets so that the original message can be reconstructed when receiving any subset of the packets when this subset is slightly higher than the size of the original message.

This paper analyses the applicability of fountain codes to the GNSS domain and its singularities and explores different implementations described in the literature. A simple implementation based on Random Linear Fountain Codes has been developed and evaluated, including transmission performance and computational resources. A comparison against traditional strategies based on offsetting and rebroadcasting is also presented for different messages such as ECDSA signature dissemination, almanac broadcasting and post-quantum signatures, in different reception conditions.

INTRODUCTION

GNSS satellites transmit their own orbit and clock information, together with some other parameters, such as ionospheric error models, in a periodic and deterministic way. Messages carrying this information are generally protected with some redundancy mechanism which provides bit-level error detection and correction, validating the correctness of the whole message. Each GNSS system implements different strategies. In the particular case of Galileo forward error correction (FEC) convolutional codes plus interleaving is used, and similar approaches are used in other new GNSS signals (e.g. LDPC for GPS L1C). FEC works in an intra-word basis, providing a layer of robustness against errors based on the redundancy of information at symbol level, hence losses of some symbols do not imply the loss of the whole word. This approach is helpful for fields of reduced length which can be codified into one or a few words. Nevertheless, there is an increasing need due to new emerging services to broadcast larger messages in an optimized, efficient and robust way, and while many future users will be internet-connected, GNSS providers cannot ignore that a big amount of users will still require standalone navigation services. This requirement is not new at all, in fact GNSS satellites are already broadcasting constellation-wide messages such as the almanacs in a non-optimized way, which implies a reception time of several minutes.

Fountain codes have been identified as a plausible candidate to solve the aforementioned problem. Fountain codes are packet coding strategies which propose an advanced encoding of the data in a set of packets to allow to the users the correct reconstruction of the message even when several packets have been lost. The main condition imposed by the fountain codes technique is that the receiver shall detect and discard the messages incorrectly received; this can be done in the GNSS navigation messages thanks to the CRC.

Given the a-priori suitability of fountain codes, a preliminary analysis to confirm whether they can be applied has been conducted. During this analysis, different fountain code implementations have been identified and assessed, selecting random linear fountain codes (RLFC) to perform a set of experimentation tests. The experimentation tests have been focused on demonstrating the suitability of the fountain codes for different applications and presenting a performance comparison with traditional approaches. Aspects like the CPU and memory resources needed to exploit the fountain codes have been also preliminarily considered in relation to commercial receiver requirements.

The work herein presented is potentially relevant due to its direct application to the real GNSS domain, both to improve existing services such as the distribution of satellites almanacs, but also for future data messages such as constellation-wide high accuracy corrections, integrity information or digital signatures.

OVERVIEW OF FOUNTAIN CODES

Fountain codes, or rateless codes, allow transforming a message into small packets so that it can be retrieved through the reception of any subset of the packets (or "drops"), as long as the number of packets received is slightly higher than the size of the original message [1]. They are based on encoding K packets into a potentially unlimited number of packets, from which N packets are received by the receiver, where $N>K$. In its simplest implementation, each of the transmitted packets ($n=1..N$) may depend on all message packets ($k=1..K$). When the receiver gathers a number of packets slightly higher than K, whichever they are, the original message can be reconstructed with a very high probability.

Fountain codes assume a channel with *erasure*: a packet is either transmitted successfully or not (erased), without any intermediate state. We can assume an erasure channel if the receiver has good error detection capabilities to know if a packet is received correctly or not. Under this assumption, fountain codes allow an effective transmission of long messages over highly noisy channels at a very low overhead.

Random Linear Fountain Codes are the simplest implementation of fountain codes. They are based on these steps:

- Divide a message into K packets: $s_1, s_2, ..., s_K$
- Define a random binary matrix G' with K rows, and a potentially unlimited number of columns. The matrix needs to be known by the transmitter and the receiver, for example by sharing a seed pseudorandom number form which a pseudorandom stream can be generated. We use a slightly different notation than [1] (G' instead of G) to differentiate between the encoding matrix (G') and the subset of it that is used by the receiver (G).
- Encode the packets as per eq. (1), where t_n is a transmitted packet, and G'_{kn} is 1 or 0.

$$t_n = \sum_{k=1}^{K} s_k G'_{nk} \quad (1)$$

- After receiving N packets, the receiver can build a G matrix with the columns of G' corresponding to the received packets, and try to invert it and determine the original packets of the message, according to eq. (2)

$$s_k = \sum_{n=1}^{N} t_n G_{nk}^{-1} \quad (2)$$

The matrix inversion requires that the N columns received are linearly independent. The probability of this is relatively high. For example, for $K \geq 10$, and a number of excess packets $E = N\text{-}K$, the probability of decoding failure $\delta(E)$ is shown in Figure 1 and bounded as:

$$\delta(E) \leq 2^{-E} \quad (3)$$

As illustrated in Figure 1, the required redundancy is very low. For example with 10 extra packets we achieve a success rate of around 99.9%. Random linear fountain codes are easy to implement and yield an acceptable computational performance, including the decoding process, which is mostly devoted to inverting G, for short messages. When K increases, the CPU consumption exponentially increases, making this approach less adequate for long messages.

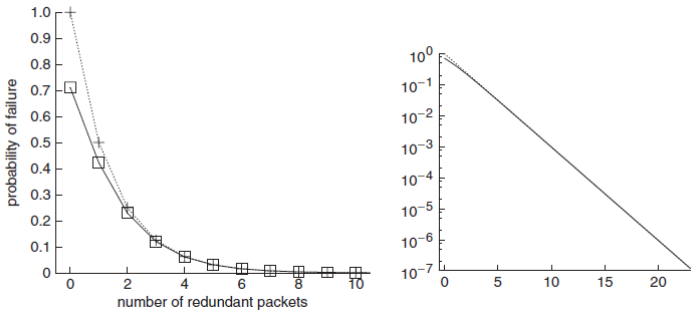


Figure 1 - Probability of failure in decoding a random linear fountain code $\delta(E)$ vs. number of redundant packets (E) [1], for a number of packets to encode K ≥10 (continuous line) vs. exponential function with base 2 (dotted line), linear (left) and logarithmic (right) scales.

FOUNTAIN CODES FOR GNSS

As a reader familiar with GNSS messages can infer, fountain codes can dramatically improve reception of long messages when an erasure channel applies, i.e. when packets are subject to a parity check with high-enough error detection capability, which is generally the case, as for example with Galileo I/NAV and its 24-bit CRC [2]. Fountain codes can be particularly useful for constellation-wide messages, and here are some present and future examples:

- Constellation-wide digital signatures or, more generally, authentication messages, e.g. for over-the-air rekeying of a private key for encrypted signals, as could be the case for Galileo Commercial Service or Open Service message authentication [3] [4]. These type of messages are particularly suited for fountain codes, as they generally cannot be used until completely received.
- Satellite almanacs.
- Constellation-wide High-Accuracy messages.
- ARAIM Integrity Support Message [5] bounding the failure probabilities and distributions of a satellite constellation.

In spite of their interesting properties, there are only few references in the open literature about the use of fountain codes for GNSS [6] [7], and none, to our knowledge, analyses in detail the performance improvement of actual GNSS message reception with realistic receiver environments and constraints.

In addition to Random Linear Fountain Codes, there are other types of fountain codes, as mentioned in [1], the main ones being LT (Luby Transform) codes, and Raptor codes. LT codes [8] follow a similar concept as random linear codes, but they use a sparser G, from which there is at least one t_n equals a s_k (i.e. a G column with a single '1'). This element represents the starting node on a graph from which the remaining original packets can be retrieved, facilitating decoding, as there is no need to invert the G matrix. It is therefore good for long (K>1000) messages. Raptor codes are evolutions of LT codes, including an outer code such as LDPC that facilitates the decoding even further. LDPC is already applied at bit level to GPS L1C data encoding but Raptor codes are covered by several patents, which may represent a barrier for their implementation in GNSS signals.

Another plausible option to ease reception of long GNSS messages, is the use of block coding as Reed Solomon [9], also proposed for GNSS, in particular for QZSS´s LEX service [10]. Reed Solomon codes can allow packet decoding when N equals K, without requiring any extra packets. Its main limitation for our purpose is that they require $N < 2^L$, where L is the packet size in bits. In other words, if very short packets are used (as e.g. the 8-bit packets proposed below), Reed Solomon are not an option.

One could combine several packets in bigger packets to counter the Reed Solomon limitation. Another solution to artificially reduce K could be dividing the target message into smaller messages coded independently, reduce decoding computational power. However, these solutions have other limitations (e.g. more redundancy and higher error rates) and they are not studied further, so the remaining sections assume the natural packet size by the constraints of the GNSS message in each implementation.

In the remaining of this paper, we have chosen Random Linear Fountain Codes for the analysis mainly because the encoding and decoding processes are easy to implement and test, and the reception performance obtained can be extrapolated to other more sophisticated coding schemes for longer messages. In addition, our main case under analysis, which is the transmission of an ECDSA signature, requires few packets (K<100), and therefore LT computational cost is within the requirements of a GNSS receiver. This does not preclude that optimizations can be performed to improve decoding.

ENCODING AN ECDSA MESSAGE WITH FOUNTAIN CODES

This section introduces a strategy to apply fountain codes to the ECDSA signature dissemination. First, the message to be encoded with RL fountain codes is defined and the approach for its codification is analysed. After that, the set of user scenarios selected to evaluate the performance and the results obtained in each user environment are presented.

The message to be encoded is 728 bits long, and is composed by a 512-bit ECDSA signature based on [11] and the signed message, which corresponds to the root key of a TESLA chain and the chain parameters, which are similar to those outlined in [4] for Galileo Open Service message authentication. Under this implementation, the Galileo message can transmit an 8-bit packet per 2-second page, which is checked by the page CRC, as in an erasure channel. The message structure (called hereinafter DSM, or Digital Signature Message) is presented in Figure 2. One can see that only 13 packets (3 to 15) transmit message information, while the rest consist of headers that need to be received only once for every signature. Under this data structure, our RL message will consist of 91 8-bit packets (K = 91), encoded in 7 Galileo 30-second subframes (13 · 7 = 91).

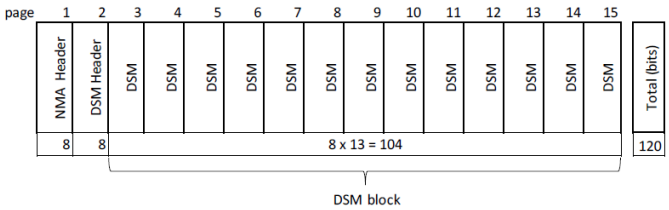


Figure 2 - OSNMA fields in the I/NAV message

One of the particularities of the proposed implementation is that, given that several satellites transmit the same message, the encoding mechanism must avoid that the same t_n is received twice, for any receiver receiving the message anytime and anywhere in the globe. In order to achieve this, the number of columns of G' has to be extended with respect to the case of only one transmitter, by approximately the number of transmitters. In our implementation, it is only extended to half the number of satellites, assuming that there is no user on Earth, or close to its surface, that can receive a signal from two satellites at opposed slots in their orbits. Another parameter required to define G', is the maximum reception time. We have assumed it to be 210 seconds or, again, 91 packets. After this time, the sequence transmitted by a satellite is repeated. While more sophisticated

approaches could be implemented, the results achieved with this approach can be considered as representative of future schemes. In the end, there are multiple ways to ensure that, whatever the N packets received, they will not be duplicated. Assuming a maximum of 32 satellites, the number of columns N' of G' is of $91 \cdot 32 / 2 = 1456$, as shown in Figure 3.

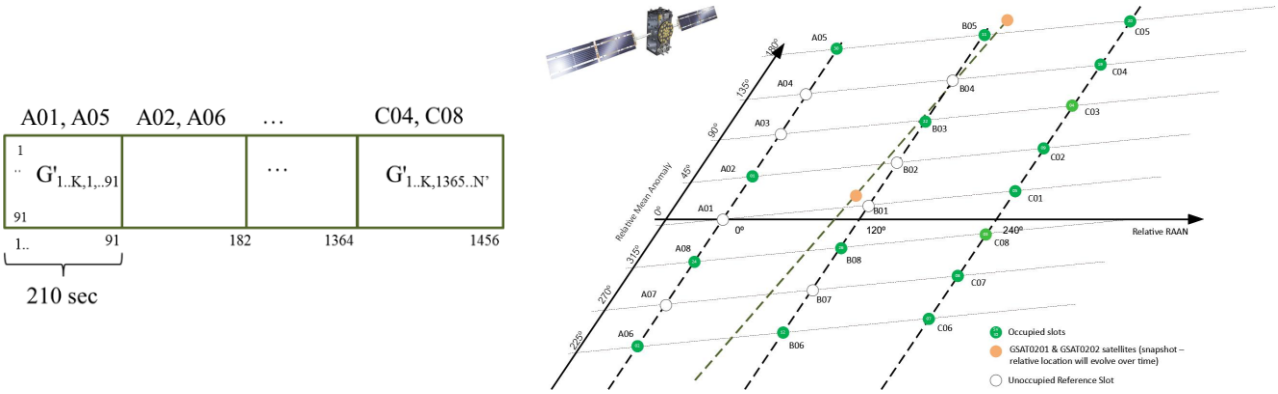
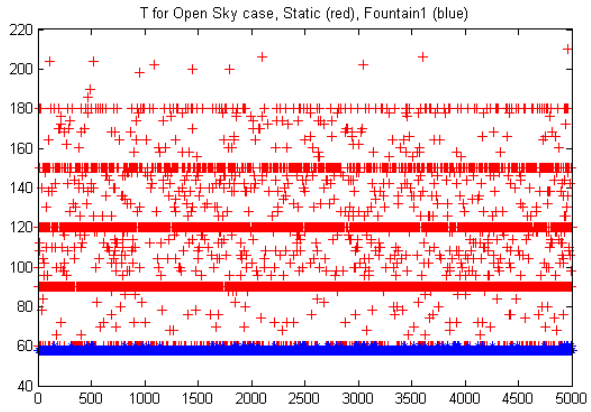
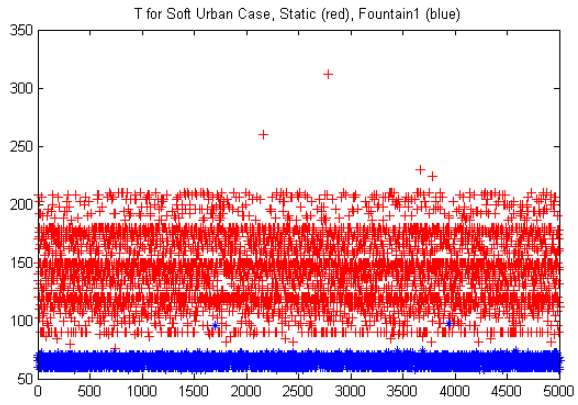
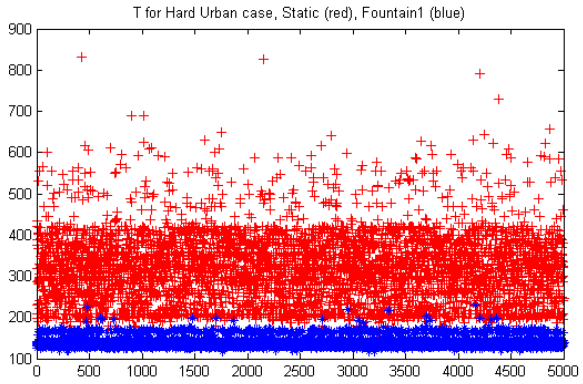


Figure 3 –G' random linear fountain code matrix as a function of time to receive the message and number of visible transmitters (left), based on current Galileo satellite slots (right, from [12])

The following step of the analysis has been to define the use cases. In order to cover different reception conditions, from harsh urban to open sky, the use cases already defined in past activities [13] have been maintained for the first part of the analysis, and then have been complemented by other scenarios, which will be described in the next sections. We recall the use case conditions in Figure 4, bottom left. The scenarios are based on realistic urban scenarios subject to Galileo uplink restrictions, and including some additional degradation. Their description is simplified to the number of satellites in view and page error rates, where page refers to a Galileo I/NAV 2-second page. The comparison is performed with the case of no coding, and static offsetting scheme as explained in detail in [13], and called 'Static' in Figure 4. This scheme applies a different transmission offset sequence to different 30-second message blocks from different satellites to reduce reception time. This approach normally behaves well when working in open-sky and low error rate scenarios, but its performance degrades significantly when in harsh user environments.







	Open Sky	Soft Urban	Hard Urban
Sats in View	4	4	2
Page Error Rates	0.5% all	1%, 5%, 10%, 20%	10%, 20%
Average Time FC[s]	58	66	139
Avg. Time NC [s]	78	127	309
99P FC [s]	60	70	174
99P NC [s]	140	192	560

Figure 4 – Scenarios and results of sequential and fountain codes applied to a digital signature message. The results (top, and bottom left), include a comparison between a sequenced approach [13] (red) and fountain codes (blue), for 5000 realizations in each environment. The bottom-right table summarizes the scenarios and results (NC = No Coding).

The figures show a dramatic improvement between the 'Static' transmission approach, and the random linear fountain encoding. The RL Fountain Codes average and 99-percentile values remain around a minute for open-sky and soft-urban cases, more than halving reception time with regard to the 'static' case, depending on the case. The hard urban case raises the difference even further: the 99-percentile improves by more than a factor of 3 (560 sec vs. 174 sec).

The next step of the analysis has been to test fountain coding with a Service Volume Emulator platform developed under the AALECS (Authentic and Accurate Location Experimentation with the Commercial Service) project. This platform is designed to emulate the Galileo System in terms of data dissemination capabilities, including available bandwidth and latency. It is able to carry out service volume simulations of the bandwidth and satellite availability which is essential to assess the data transmission capabilities under different scenario configurations. The Service Volume Simulator is highly configurable and allows the user to emulate a realistic Galileo System including:

- Galileo constellation: satellite ephemeris, number of satellites, etc.
- Ground segment: number and location of the uplink stations and antennas.
- Worldwide coverage: map grid with the location of users all around de globe.
- Reception conditions: user location, masking angle, bit/page error model based on satellite elevations.
- Mission Uplink plan: satellite connection status using real Galileo contact plan as input.

Thanks to these features, the Service Volume Emulator offers a useful framework to evaluate new services that might be offered in the future to the GNSS community. In this sense, the Emulator has been adapted to support the possibility of analysing the reception and dissemination of the Digital Signature Message as per the abovementioned specification. It supports DSM dissemination analysis both in plain format ('Static' case abovementioned) and encoding the DSM using the RL Fountain Codes. Several simulations have been conducted and the performance of both approaches have been analysed. The simulations have been performed with the following configurations:

- Galileo constellation: FOC constellation, 24 satellites with realistic orbital parameters.
- Ground segment: 5 stations with 4 uplink antennas each (expected FOC configuration).
- Worldwide coverage: worldwide map simulated with a grid of 20-degree cells.
- Receiver configuration:
  - Open sky: 5 degrees masking angle and no reception errors.
  - Urban (LMS-based): 30-degree masking angle and errors introduced based on the Land Mobile Satellite model [14]. Urban (real error-based): 30-degree masking angle and errors injected based on real recordings of Galileo E1B data in urban environments. The bit error rates in this case are larger than in the LMS one.

Figure 5, Figure 6 and Figure 7 depict the results of the simulations performed both for the 'static'/no coding/sequenced approach and fountain codes.

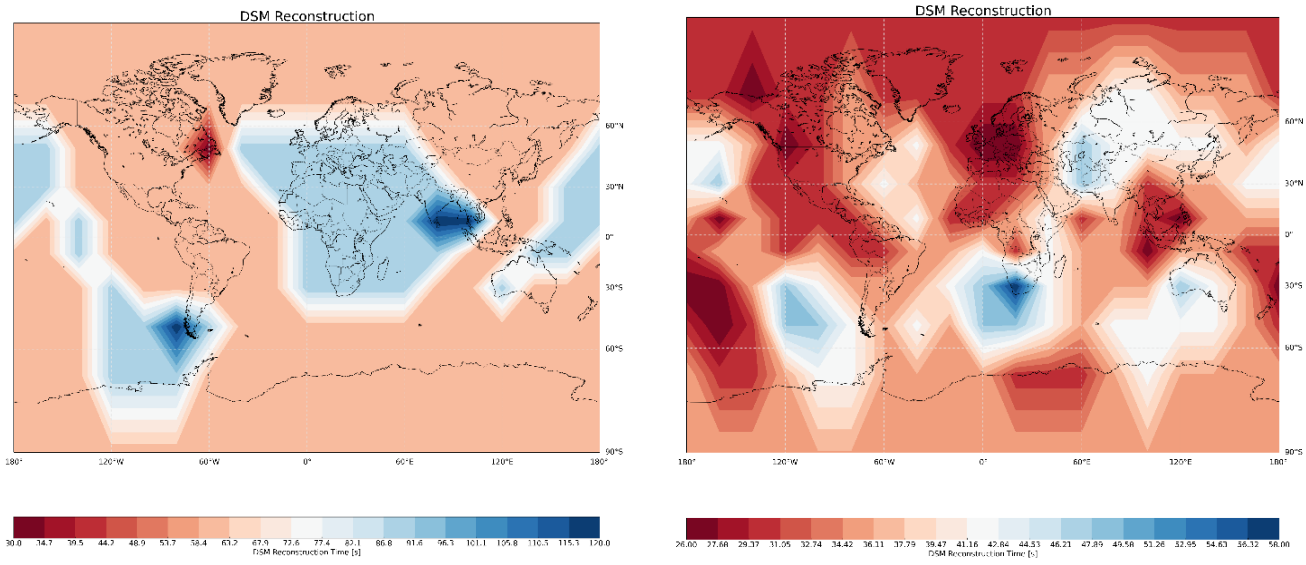


Figure 5 - Open Sky no coding (left) vs. fountain codes (right)

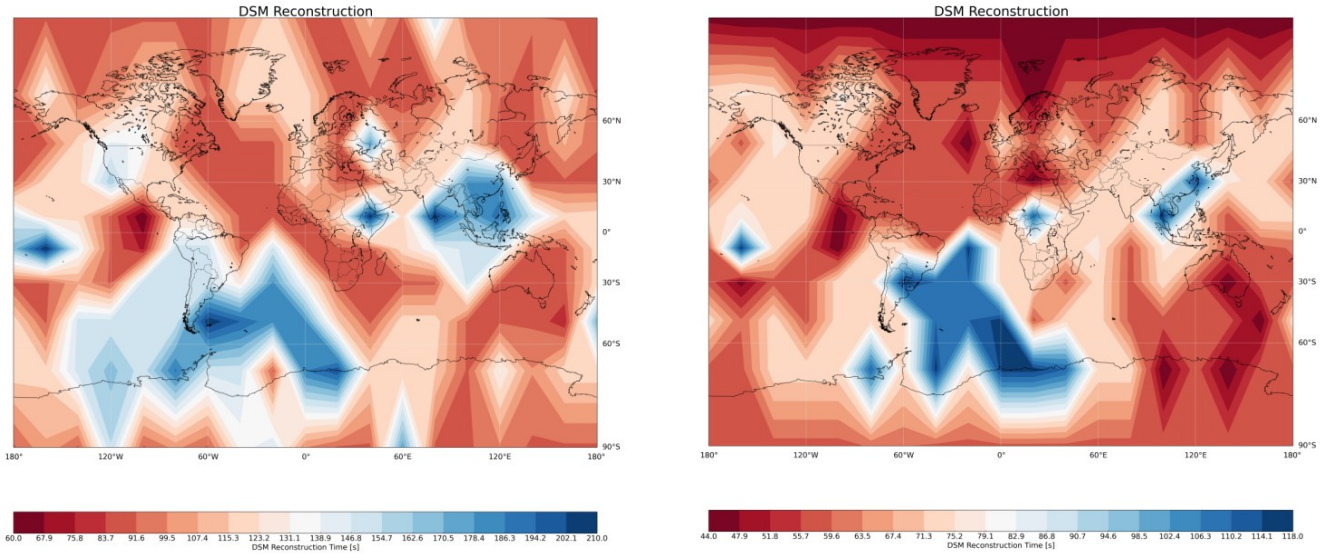


Figure 6 - Urban LMS, no coding (left) vs. fountain codes (right)

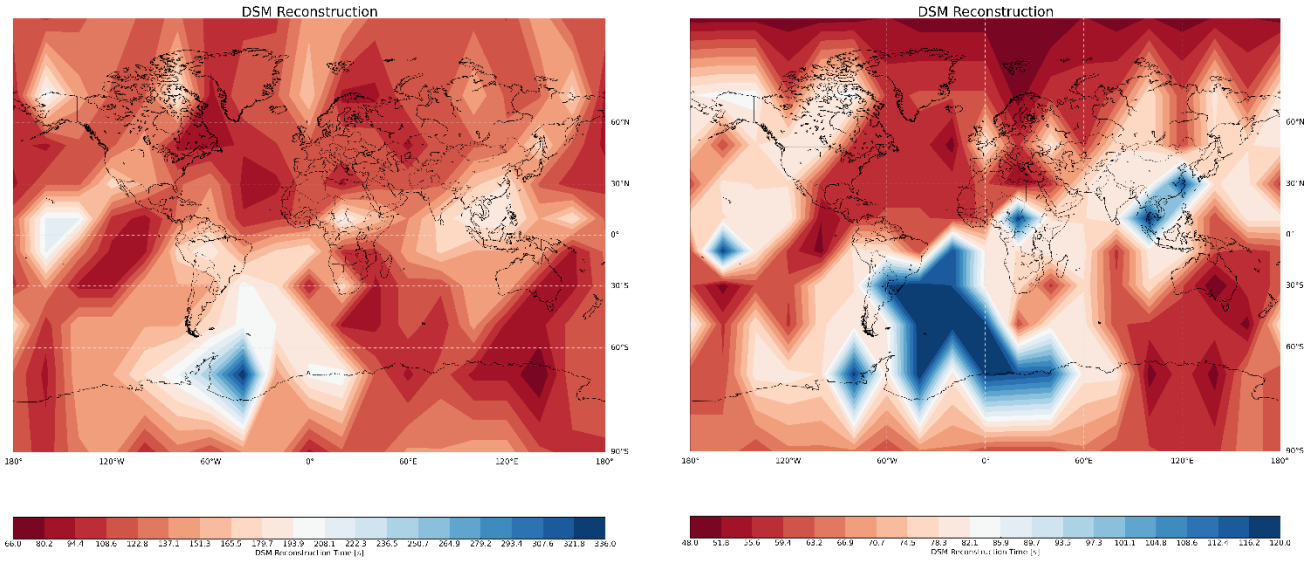


Figure 7 - Urban with real errors, no coding (left) vs. fountain codes (right)

As it can be observed in the results, the impact of using the FC encoding is very positive, especially in harsh user conditions. For the worst cases, i.e. when less satellites are available and more pages are lost, we can see the great improvement of using FC as the reception time reduces in more than half. In particular, when assessing the worst case results, both LMS and real error models, it can be also concluded that a high increase on the error rates has less impact on the FC implementation than on the sequenced approach.

Table 1 - Results summary

	Open Sky	**Urban (LMS)**	**Urban (Real)**
Best case (no coding)	30s	60s	66s
Best case (FC)	26s	44s	48s
Worst case (no coding)	120s	210s	336s
Worst case (FC)	58s	118s	120s

COMPUTATIONAL COST

In order to confirm whether fountain codes are a plausible option for GNSS data encoding, its performance shall be also assessed in terms of computational resource requirements. A specific test to analyze the CPU time needed to process the received and decode an ECDSA signature has been performed with the RL implementation previously presented. It shall be noted that the RL is considered one of the most CPU demanding options within the fountain codes, therefore other options should behave better in terms of CPU load. The platform used for the execution of this test is a laptop with an Intel Core i7 CPU running at up to 2.4GHz. The RL implementation has been developed in Matlab, which is considered a non-optimized development language so there is also margin to obtain better results employing more efficient programming languages such as C, C++ or Java. Figure 8 shows the results obtained in 500 executions of the decoding process. As it can be observed 99% of the execution the time required is below 0.05 seconds and normally below 0.04 seconds. Extrapolating this figure to a receiver CPU with few hundreds MHz, the processing time would be in the range of 0.3-0.5 seconds, before using a more optimized language such as C, C++ or Java, which may reduce this time significantly. Regarding the memory needed, assuming the worst case of storing the G' matrix in memory instead of generating it using a keystream generator the required memory is 16 Kbytes, hence it can be considered low. In any case, this preliminary analysis shows that random linear fountain code decoding algorithms for short (K<100) messages are affordable for nowadays' and future GNSS receivers.

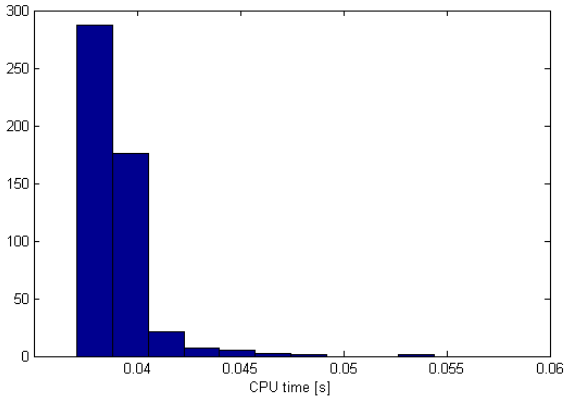


Figure 8 – CPU time required for ECDSA RL decoding

GALILEO ALMANACS

Another potential candidate to adopt fountain codes is the satellite almanac message. Almanacs are normally broadcast by all the satellites and the time to receive the full almanac is relatively long. In the particular case of Galileo, the transmission takes 12 minutes being only two pages within the whole I/NAV 30-second subframe intended for the almanacs data, hence only 48 pages contain almanac information. The approach proposed by fountain codes can be applied to this scenario as follows: each GNSS satellite represents one fountain and each page disseminated through the SIS represents one packet (or drop). With this configuration each satellite may be broadcasting two different drops in each subframe. The number of drops gathered by a receiver may depend on the satellite visibility i.e. end-user conditions. The same three scenarios analyzed in the ECDSA case using the service volume simulator are also used in this analysis (Open-sky with 5º masking angle, LMS with 30º masking angle, and Real urban with 30º masking angle). Figure 9 presents the open sky scenario results, which provides an idea of the highest improvement which can be achieved with the proposed scheme: the time to reconstruct the full-almanac improves from the 12 minutes required with the current data structure, to 2 minutes in most locations, with a maximum of 210 seconds in some specific locations.

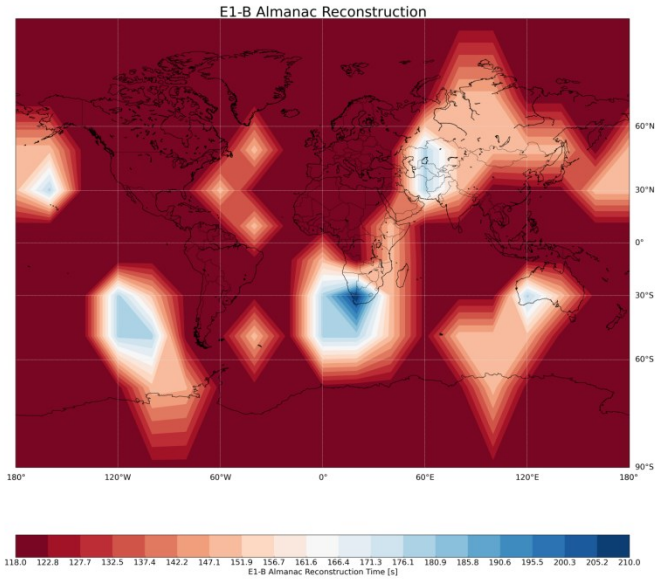


Figure 9 - Almanacs reconstruction time in open-sky scenario

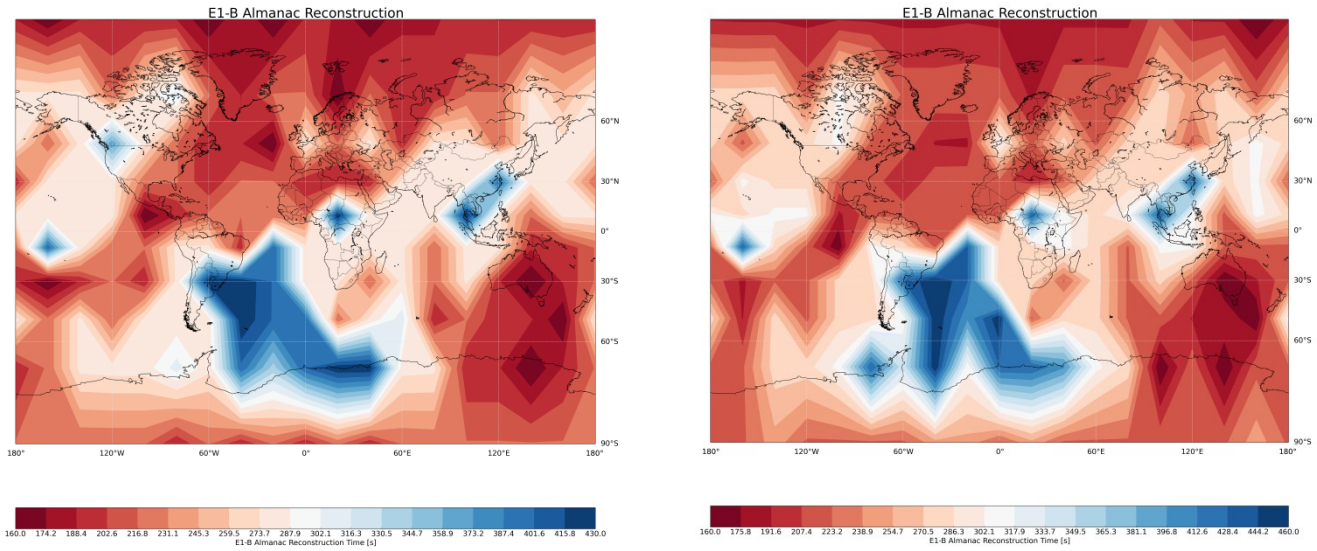


Figure 10 - Almanacs reconstruction time in LMS scenario (left) and real urban errors scenario (right)

Figure 9 results are obtained in an open sky scenario without introducing transmission errors, which may be considered benign conditions for the fountain codes since no drop is lost and the reconstruction is executed in an optimal way. These results are complemented with the two more challenging analyses abovementioned, which are representative of harsh urban conditions i.e. high masking angle and high frame error rates, and are presented in Figure 10. The outcome of this set of executions is comparable between them, and also confirms that the packet-oriented dissemination strategy based on fountain codes reduces the time of reception of the almanac from the current minimum of 720 seconds to a range of 160-460 seconds in challenging scenarios.

POST-QUANTUM DIGITAL SIGNATURES

Due to the foreseen emergence of the quantum computers, common digital signatures as we know them (RSA, ECDSA, …) may become vulnerable to cryptanalysis attacks: a quantum computer will be able to solve the integer factorization and discrete logarithm problem in polynomial time, which are the mathematical problems on which the classic asymmetric cryptography relies. New post-quantum cryptography is a topic that is being widely investigated. So, not only the cryptographic protocols are going to change but also the asymmetric keys and signature sizes may suffer a dramatic increase in the future to secure the authentication operations. Currently, post-quantum signatures sizes are in the order of kilobytes (~1 KB to 40KB). This increase on signature sizes makes more important to explore ways to optimize the transmission of large sets of data through erasure channels with limited bandwidth, as it the case for GNSS satellites.

There are several promising post-quantum signatures being researched at present time, as e.g. hash-based [15], code-based, or lattice-based. For the present analysis, a digital signature defined in [16] called Dilithium has been selected. Dilithium is a lattice-based signature scheme whose signature has a size of 2.7 Kilobytes. The selected signature is several times longer than any asymmetric-based signature size being used at the moment in classical cryptography, and is representative of the signature sizes that are going to be required to provide protection against future crypto-attacks with quantum computers.

Two tests using the service volume platform have been conducted: a first one aiming to obtain the transmission time of the signature without any packet-based codification and just establishing a static block offset sequence for the transmitting satellites in a random manner; and the second one employing the codification using the Random Linear Fountain Codes approach previously proposed. For the conducted analysis error-free conditions and 5º masking angle scenario is tested, using only a 4 bps per satellite configuration, as in the ECDSA case. Figure 11 shows that the best case for the reception of the digital signature encoded using the fountain codes is ~11 minutes, and reception time mostly remains below 15 minutes. This may be acceptable for future users if post-quantum signatures are only seldom required and the receiver has a long time span to retrieve the message (otherwise a higher bitrate may be necessary); on the other hand using a plain transmission plus offsetting approach requires more than half an hour in almost all locations. This represents an improvement of more than 65% when using the packet-encoding strategy based on fountain codes.

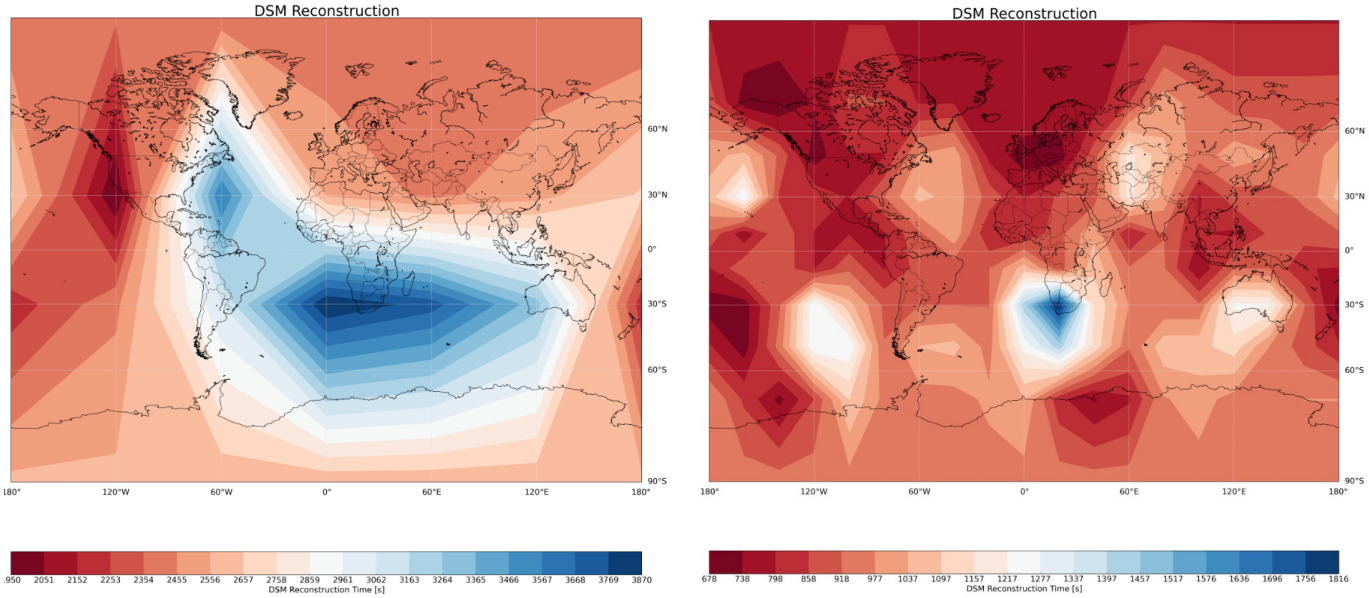


Figure 11 - Post-Quantum signature Service Volume results: no coding (left) vs. Fountain codes (right)

CONCLUSION

GNSS data is encoded in the signals through different strategies to improve reception robustness. For example, Galileo employs convolutional codes with interleaving to increase the probability of word reconstruction in the receiver. This technique works at symbol/bit level and the benefit is only observed at word-level. Nevertheless, there is an increasing need of improving the dissemination of large messages which are normally distributed using several words/pages of the navigation message due to bandwidth constraints. A main problem found for the distribution of these messages is that losing only one chunk makes the reconstruction of the message impossible until it is rebroadcast. The presented work has explored and demonstrated the use of fountain codes for packet-oriented transmission, in order to provide enhanced robustness and redundancy to improve the reconstruction of the transmitted messages, and its suitability for GNSS data encoding.

An ad-hoc implementation based on Random Linear fountain codes has been developed and tested. Transmission performance has been analyzed for different conditions and three different messages where the fountain codes may be directly applied: an ECDSA signature message for Galileo OS message authentication, Galileo almanacs, and a long, post-quantum digital signature. A dedicated fountain code message decoding CPU load analysis has been conducted for the ECDSA case, and the results show that the required processing time in a Matlab, non-optimized implementation running in a commercial laptop are below 50 milliseconds, making it affordable in commercial receivers, assuming that decoding the ECDSA signature is not a frequent operation.

Fountain codes have been compared with an offset-based dissemination strategy, already optimized to minimize message reception time, but without an extra coding layer. The results show a dramatic improvement of the reception performance in all the scenarios analyzed with respect to the nominal dissemination strategy, generally by a factor of 2 or 3, depending on the user conditions. This improvement is even higher for current, non-optimized message transmissions, as is the case of the Galileo almanacs. We can therefore conclude that fountain codes are good candidates for improve the reception and decoding performance for long messages broadcast by GNSS systems.

ACKNOWLEDGMENTS

The authors would like to thank J. Simón, Prof. V. Rijmen and all the members who have actively participated in the AALECS project.

REFERENCES

[1] D. MacKay, "Fountain Codes," *IEEE Proc.-Commun. - Capacity Approaching Codes Design and Implementation Special Section,* vol. 152, no. 6, 2005.

[2] European Union, "Galileo OS SIS ICD: Open Service Signal In Space Interface Control Document v1.2," European Union, Nov 2015.

[3] D. Calle, S. Cancela, E. Carbonell, I. Rodríguez, G. Tobías and I. Fernández-Hernández, "First Experimentation Results with the Full Galileo CS Demonstrator," in *ION GNSS+ 2016*, Portland, OR, 2016.

[4] I. Fernández-Hernández, V. Rijmen, G. Seco-Granados, J. Simón, I. Rodríguez and J. D. Calle, "A Navigation Message Authentication Proposal for the Galileo Open Service," *NAVIGATION, the Journal of the Institute of Navigation,* 2016.

[5] J. Blanch, T. Walter, P. Enge, S. Wallner, F. Amarillo, R. Dellago, R. Ioannides, I. Fernandez, B. Belabbas, A. Spletter and M. Rippl, "Critical Elements for a Multi-Constellation Advanced RAIM," vol. 60, no. 1, pp. 53-69, 2013.

[6] M. Wu, Q. Yu and W. Meng, "Application of Fountain Code to GPS Navigation Data Structure Design," in *7th International Conference on Wireless Communications, Networking and Mobile Computing (WiCOM)*, Wuhan, China, 2011 .

[7] Q. Liu, W. Zhang, Y. Wang and H. Li, "The Application of Fountain Code in Satellite Navigation System," in *China Satellite Navigation Conference (CSNC) 2016 Proceedings:, Volumen 2*, 2016.

[8] M. Luby, "LT codes," Proceedings of the 43 rd Annual IEEE Symposium on Foundations of Computer Science (FOCS'02).

[9] A. Goldsmith, "Wireless Communications," Cambridge University Press, 2005.

[10] S. Choy, K. Harima, Y. Wakabayashi, S. Kogure, Y. Li, M. Choudhury and C. Rizos, "High Accuracy Real-Time Precise Point Positioning using the Japanese Quasi-Zenith Satellite System LEX Signal," in *Geospatial Science Research*, 2014.

[11] National Institute of Standards and Technology, "FIPS PUB 186-4 - Digital Signature Standard (DSS)," U.S. Department of Commerce, 2013.

[12] European Union, "European GNSS Service Centre," [Online]. Available: https://www.gsc-europa.eu/system-status/orbital-and-technical-parameters. [Accessed 27 07 2017].

[13] I. Fernández-Hernández, J. D. C. Calle, S. Cancela, O. Pozzobon, C. Sarto and J. Simón, "Packet transmission through navigation satellites: A preliminary analysis using Monte Carlo simulations," in *ENC 2017*, Lausanne, CH, 2017.

[14] F. P. Fontan, M. Vázquez-Castro, C. E. Cabado, J. P. Garcia and E. Kubista, "Statistical modeling of the LMS channel," *IEEE Transactions on Vehicular Technology,* vol. 50(6), pp. 1549-1567, 2001.

[15] D. J. Bernstein, D. Hopwood, A. Hülsing, T. Lange, R. Niederhagen, L. Papachristodoulou, M. Schneider, P. Schwabe and Z. Wilcox-O'Hearn, "SPHINCS: practical stateless hash-based signatures.," in *Eurocrypt 2015*, 2015.02.02.

[16] L. Ducas, T. Lepoint, V. Lyubashevsky, P. Schwabe, G. Seiler and D. Stehlé, "CRYSTALS – Dilithium: Digital Signatures from Module Lattices.," 2017-06-27.