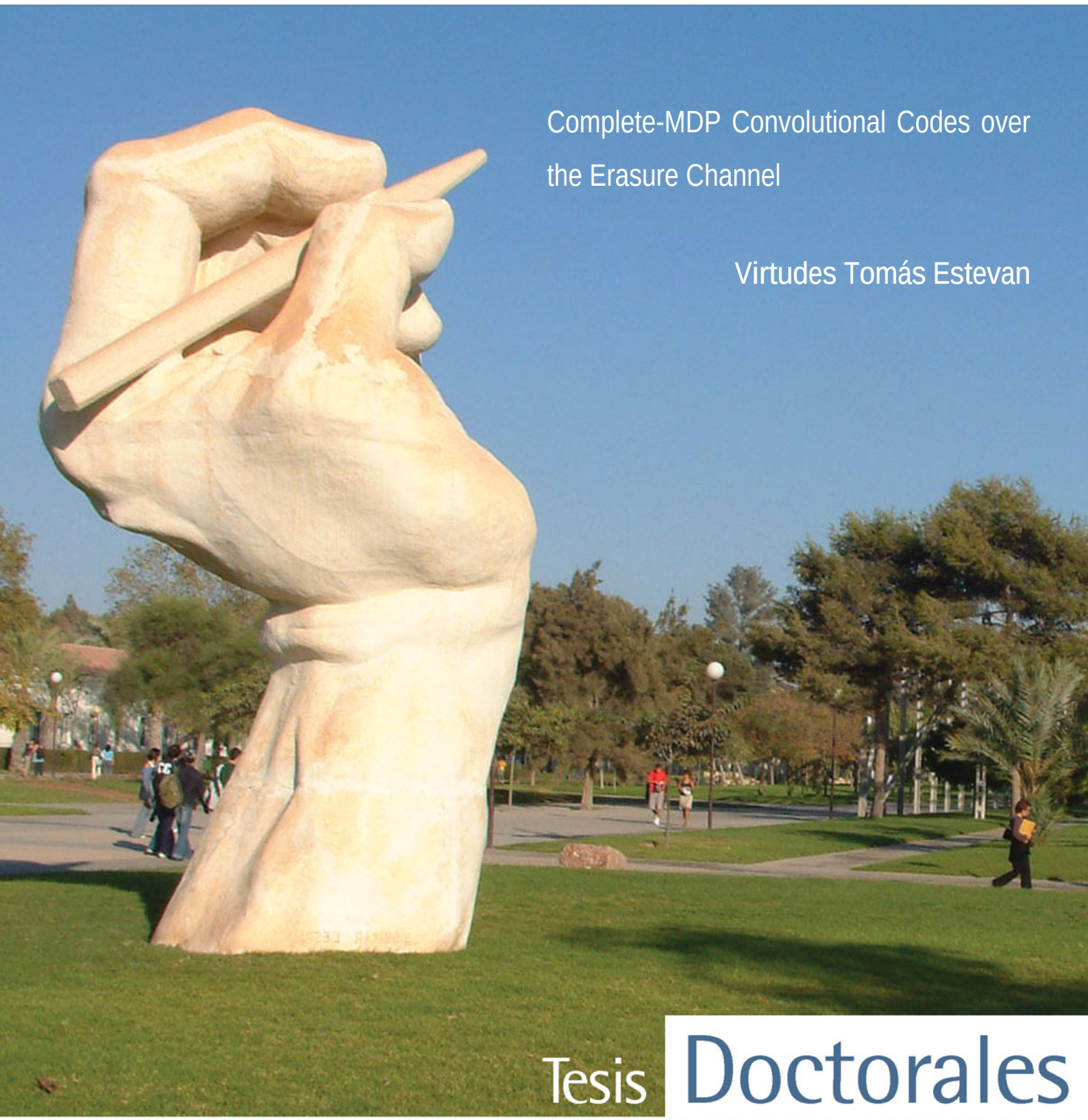
Universitat d'Alacant
Universidad de Alicante

Complete-MDP Convolutional Codes over the Erasure Channel

Virtudes Tomás Estevan



Tesis Doctorales

www.eltallerdigital.com

UNIVERSIDAD de ALICANTE

Universidad de Alicante

Departamento de Ciencia de la Computación e Inteligencia Artificial

Universitat d'Alacant
Universidad de Alicante

Complete-MDP Convolutional Codes over the Erasure Channel

TESIS DOCTORAL

Presentada por:
Virtudes Tomás Estevan

Codirigida por:
Joan Josep Climent Coloma
Joachim Rosenthal

Universitat d'Alacant
Universidad de Alicante

Universidad de Alicante

Departamento de Ciencia de la Computación e Inteligencia Artificial

Complete-MDP Convolutional Codes over the Erasure Channel

Universitat d'Alacant
Universidad de Alicante

Memoria presentada para optar al grado de doctora por Virtudes Tomás Estevan.

Alicante, junio de 2010.

Universitat d'Alacant
Universidad de Alicante

D. Joan Josep Climent Coloma, Catedrático de Universidad del Departamento de Estadística e Investigación Operativa de la Universidad de Alicante y D. Joachim Rosenthal, Catedrático del Departamento de Matemáticas de la Universidad de Zürich

CERTIFICAN:

Que la presente memoria *Complete-MDP Convolutional Codes over the Erasure Channel*, ha sido realizada bajo su dirección, en el Departamento de Ciencia de la Computación e Inteligencia Artificial de la Universidad de Alicante por la licenciada Dña. Virtudes Tomás Estevan, y constituye su tesis para optar al grado de Doctora.

Para que conste, en cumplimiento de la legislación vigente, autorizan la presentación de la referida tesis doctoral ante la Comisión de Estudios de Posgrado de la Universidad de Alicante, firmando el presente certificado.

Universitat d'Alacant
Universidad de Alicante

Alicante, 1 de junio de 2010

Joan Josep Climent Coloma

Joachim Rosenthal

Universitat d'Alacant
Universidad de Alicante

A mis padres,
y a Michael.

Universitat d'Alacant
Universidad de Alicante

Universitat d'Alacant
Universidad de Alicante

Agradece a la llama su luz, pero no olvides el pie del candil que,
constante y paciente, la sostiene en la sombra.
Rabindranath Tagore

Este "gracias" va dirigido tanto a las llamas que alumbraron mi camino como a los candiles que me apoyaron en él. Profesores, familia y amigos han estado siempre ahí.

En primer lugar, quiero agradecer a mis directores de tesis Joan Josep Climent y Joachim Rosenthal todo el apoyo que me han dado a lo largo de este tiempo. Sin su ayuda, conocimientos e ideas no podría haber llevado acabo este trabajo. Siempre alentadores, me animaron a creer en mí misma y a levantarme de nuevo en cada tropiezo. Con su constancia y respaldo aprendí a ser "tonta" una y otra vez, porque, como tantas veces me dijo Joan Josep, para poder investigar hay que serlo, pues sólo los tontos son capaces de ilusionarse cien veces.

Gracias al grupo de Álgebra Aplicada de la Universidad de Zürich y, en especial, a mi codirector Joachim Rosenthal, por su calurosa acogida y por brindarme la oportunidad de trabajar con ellos durante las estancias que allí realicé.

Me gustaría dar las gracias por su gran hospitalidad durante mi visita a San Diego a Roxana Smarandache, de quien he aprendido mucho. Gracias por animarme siempre a seguir superándome, gracias por todas las conversaciones en los buenos y malos momentos.

A todos mis amigos y compañeros quiero también decirles gracias. Gracias por compartir dudas, discusiones, comidas y cafés, y, por supuesto, papeleos y burocracias, Vero y Sara saben a qué me refiero. Gracias a aquellos que mostraban su interés preguntando aún sin entender, gracias por estar ahí cuando he necesitado reir o llorar, gracias por acompañarme en tantos momentos.

Y como no, ante todo quisiera darle las gracias a mi familia y a los que ahora también son mi familia, a los que me han acompañado en la distancia y al que ha estado ahí cada día al regresar a casa. Gracias por estar a mi lado apoyándome y respaldándome a lo largo del camino. ¡MUCHAS GRACIAS!

Universitat d'Alacant
Universidad de Alicante

Universitat d'Alacant
Universidad de Alicante

Contents

Universitat d'Alacant
Universidad de Alicante

Universitat d'Alacant
Universidad de Alicante

Prólogo

Hoy en día, conseguir una comunicación fiable y eficiente se ha convertido en uno de los problemas centrales para la sociedad de la información en que vivimos. La transmisión de información implica el uso de canales de comunicación. Tanto si nos comunicamos a través del teléfono, de Internet, por ondas de radio o directamente de persona a persona, utilizamos canales que sufren los efectos de ruidos y perturbaciones. Esto afecta a los mensajes que enviamos y por ello, la información correctamente generada por el emisor puede llegar distorsionada al receptor. La teoría de códigos aparece como solución a este problema. Por una parte, los códigos detectores y correctores de errores determinan si el mensaje recibido difiere del que fue enviado originalmente, y por otra parte, intentan recuperar la información original a partir de la secuencia recibida, posiblemente alterada.

En el proceso de codificación, cierta información adicional o *redundancia* es añadida al mensaje original de forma óptima de modo que cualesquiera dos mensajes son codificados de forma substancialmente diferente. Así, bajo la presencia de un número relativamente pequeño de errores, el receptor es capaz de diferenciar entre dos mensajes distintos, y más aún, los errores pueden ser detectados, y en muchos casos, corregidos.

El esquema del proceso de comunicación se puede ver representado en la figura 1. En primer lugar, el emisor genera la información que desea transmitir y la envía al codificador. En el codificador la información es dividida y transformada en palabras código siguiendo el método de codificación asociado al código concreto que se esté utilizando. Después, las palabras código son enviadas a través del canal y llegan al decodificador; estas pueden verse afectadas por errores debido al ruido en la transmisión. El dispositivo de decodificación analiza el mensaje recibido e intenta determinar si han ocurrido errores en la comunicación. Cuando estos ocurren, trata

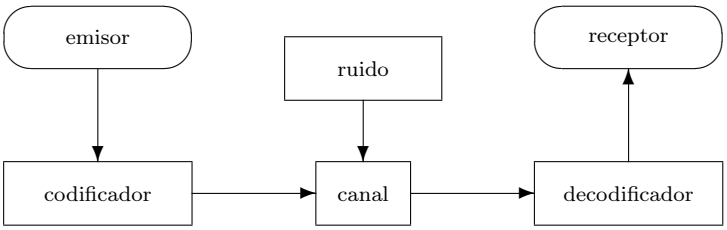


Figura 1: *Proceso de comunicación*

de corregirlos para obtener las palabras código originales que fueron enviadas. Después invierte el proceso de codificación y recupera el mensaje original. Finalmente, el mensaje se envía al receptor.

Gracias a la teoría matemática sobre la que se han desarrollado los códigos correctores, actualmente podemos disfrutar de muchos dispositivos y sistemas digitales, como son las memorias RAM de los ordenadores, los CD, los discos duros, la comunicación inalámbrica, los sistemas de sonido digital, la comunicación vía satélite, etc.

Los códigos detectores y correctores de errores se pueden dividir fundamentalmente en dos clases: códigos bloque y códigos convolucionales. Los códigos bloque dividen la información en bloques de longitud fija que después son codificados de forma independiente y siguiendo continuamente el mismo procedimiento. Además, estos códigos transforman siempre un cierto bloque de información en la misma palabra código. Es por ello que podemos interpretar un código bloque como una caja negra con un mecanismo fijo, en la que la información es transformada utilizando siempre el mismo algoritmo. La estructura algebraica de los códigos bloque ha sido ampliamente estudiada y se sabe que tienen muy buena capacidad correctora y algoritmos de decodificación algebraicos fáciles de implementar.

Los códigos convolucionales, sin embargo, consideran la información como una secuencia completa. Aunque también dividen la información en bloques de una longitud fija, tienen en cuenta la posición relativa de cada bloque en la secuencia. Además, los bloques no se codifican de forma independiente, sino que bloques de información anteriores afectan a bloques que se codifican posteriormente, es decir, existe solapamiento. De este modo, al contrario que en el caso de los códigos bloque,

una cierta información no se codifica siempre como una palabra código determinada, ya que esto depende de cuál es su posición en el mensaje completo. A pesar de que en los últimos años muchos autores se han interesado por el estudio de la estructura algebraica de los códigos convolucionales, hay muchos aspectos que no se conocen tan profundamente como en la teoría de códigos bloque.

Los códigos convolucionales fueron introducidos en 1955 por Elias [20] y desde entonces se han estudiado bajo distintos puntos de vista. Utilizando nociones de álgebra lineal, se pueden considerar como subespacios vectoriales del cuerpo de las series de Laurent (véase [26, 42, 59, 65]). Según la teoría de sistemas dinámicos simbólicos, son subespacios vectoriales, compactos, irreducibles e invariantes por desplazamientos del anillo de las series de potencias formales $\mathbb{F}[[z, z^{-1}]]$ (véase [48, 53, 57]). Otros autores como Willems (véase [85, 86, 87]) los definen como trayectorias o comportamientos (del término en inglés, *behavior*) completos, lineales e invariantes en el tiempo. Si se trabaja desde el punto de vista de módulos obtenemos la representación introducida por Fornasini y Valcher [25, 81] y Rosenthal y otros autores [71, 73, 84] donde un código convolucional es considerado como un submódulo del módulo $\mathbb{F}^n[z]$. Esta caracterización basada en la teoría de módulos requiere que las palabras código tengan soporte finito y está relacionada con los mencionados *behaviors* mediante la dualidad de Pontryagin (véase [70]). Bajo el punto de vista de la teoría de sistemas lineales, existen varias representaciones de primer orden asociadas a los códigos convolucionales. En esencia, se considera un sistema lineal para describir el comportamiento del código convolucional a lo largo del tiempo. Kalman y algunos de sus coautores [43, 44] demostraron que la función de codificación se puede factorizar como una realización de la matriz generadora del código. Más tarde, Fuhrmann [28] refinó este proceso de realización. Utilizaremos varias de estas representaciones a lo largo de la tesis.

Este trabajo está organizado como sigue. Dedicamos el capítulo 1 a la introducción de los conceptos básicos necesarios para el entendimiento del resto de la memoria. En primer lugar, presentamos algunos preliminares sobre códigos bloque. Estos no son el tema central de esta tesis, pero aparecen en numerosas ocasiones al ser comparados con los códigos convolucionales, tanto por su comportamiento sobre canales de borrado, como por las analogías en sus estructuras. Las secciones 1.3 y 1.4 presentan las diferentes representaciones de códigos convolucionales que utilizamos en el resto de capítulos. La primera es la descripción basada en la teoría

de módulos donde los códigos convolucionales son considerados como submódulos del módulo $\mathbb{F}^n[z]$. Utilizamos esta descripción a lo largo del capítulo 3. La segunda representación está basada en la teoría de sistemas lineales y es conocida como la representación entrada-estado-salida o representación (A, B, C, D). En este caso, las matrices (A, B, C, D) describen un sistema lineal que gobierna la evolución del código convolucional en el tiempo. Esta descripción aparece en los capítulos 4 y 5 de la memoria.

En el segundo capítulo estudiamos la generación de matrices superregulares y superregulares-reversas. Las matrices superregulares son un tipo especial de matrices necesarias para la construcción de códigos convolucionales con perfil de distancia máximo (MDP). Son matrices triangulares inferiores con estructura Toeplitz, de dimensión finita y con elementos sobre cuerpos finitos, que tienen la propiedad de que todos los menores de cualquier tamaño que tienen la posibilidad de ser distintos de cero, es decir, que no incluyen demasiados ceros por encima de la diagonal, son distintos de cero. Estos son llamados *menores no trivialmente nulos*. Los códigos convolucionales MDP son una clase de códigos con muy buena capacidad correctora por intervalo de tiempo, gracias a que las matrices de paridad asociadas a los mismos en cada instante son equivalentes a las matrices de paridad de un código bloque con máxima capacidad correctora, esto es, un código bloque de máxima distancia separable (MDS). Esta optimalidad se debe a que cualquier menor de tamaño máximo de las matrices de paridad tiene rango completo. En las construcciones de códigos MDP presentadas hasta ahora, los menores no trivialmente nulos distintos de cero de matrices superregulares se traducen directamente en que las matrices de paridad mantienen el rango completo de sus menores de tamaño máximo. Esta relación directa convierte a las matrices superregulares en una pieza clave para la construcción de códigos MDP.

Sin embargo, poco se conoce sobre la construcción directa de estas matrices. Existe un método conocido para la generación de matrices superregulares sobre $\mathbb{F}_p$ (véase [33]), pero la cardinalidad del cuerpo empleado es demasiado grande, siendo además imposible predecir el tamaño de cuerpo mínimo necesario para construir una matriz de tamaño $n \times n$. Por ello es necesario el desarrollo de métodos directos de construcción de matrices superregulares que mantengan un tamaño de cuerpo menor evitando así las exhaustivas búsquedas por ordenador, cuya complejidad es elevada.

Universitat d'Alacant
Universidad de Alicante

En esta memoria analizamos las matrices superregulares en conexión con la teoría de secuencias totalmente positivas. Edrei, junto con otros autores [2, 17, 18], introdujeron este concepto en los años 50. Las secuencias totalmente positivas tienen la capacidad de generar matrices de números reales cuyos menores de cualquier orden son no negativos. Dichos autores caracterizaron los tres tipos de factores que generan estas secuencias. En nuestro trabajo consideramos los polinomios generadores de matrices superregulares como una subclase de las secuencias totalmente positivas. De esta forma, utilizando los mismos tipos de factores podemos generar estas matrices. En la sección 2.2, presentamos un método para la construcción de matrices superregulares basado en la elección de un generador de $\mathbb{F}_p$ o de un generador de $\mathbb{F}_{p^n}$ que depende del polinomio irreducible utilizado para definir dicho cuerpo a partir de $\mathbb{F}_p$ (el único conocido hasta el momento sobre este tipo de cuerpos), así como diversas transformaciones que mantienen la superregularidad.

Sin embargo, las condiciones suficientes propuestas para secuencias totalmente positivas de números reales no son directamente transferibles al caso de polinomios generadores con coeficientes en un cuerpo finito. Esto es debido a que la aritmética sobre cuerpos finitos no satisface las mismas propiedades que la aritmética de números reales, como por ejemplo, que la suma de dos números positivos es distinta de cero, o que el concepto de límite no está claramente definido. Es por ello que la elección correcta del elemento generador o del polinomio irreducible no puede ser hecha *a priori*.

A pesar de esto, nuestro método reduce drásticamente las búsquedas exhaustivas ya que no es necesario comprobar cualquier combinación de elementos en el cuerpo, sino simplemente comprobar los diferentes elementos generadores o polinomios irreducibles, cuyo número es mucho inferior. Además, el tamaño mínimo necesario para nuestra construcción es menor que el obtenido en [33]. Junto con las transformaciones que conservan la superregularidad y proporcionan nuevas matrices, este método se convierte en una potente herramienta para generar matrices superregulares.

En este capítulo introducimos también un nuevo concepto: matrices superregulares-reversas. Estas son matrices que siguen siendo superregulares cuando la posición de sus elementos está invertida rescepto a una determinada diagonal inferior. Las matrices superregulares-reversas juegan un papel importante en la construcción de los llamados códigos convolucionales MDP-reversos, que constituyen un nuevo tipo de códigos convolucionales y son introducidos en el capítulo 3. Las propiedades par-

ticulares de estas matrices hacen que su búsqueda sea más complicada. Sin embargo, en la sección 2.3 probamos que, de hecho, nuestra construcción genera este tipo de matrices y presentamos transformaciones que preservan la superregularidad-reversa. Esto pone de nuevo de manifiesto la utilidad del método aquí presentado.

Por último, hacemos una comparativa de los tamaños de cuerpo necesarios en cada caso y detallamos, para matrices de tamaño hasta 9×9, los elementos generadores y polinomios irreducibles que podemos utilizar para el cálculo.

En el capítulo 3 estudiamos la decodificación de códigos convolucionales sobre canales con borrado como alternativa a los códigos bloque. Debido a su mayor difusión nos referiremos al canal con borrado con el término en inglés *erasure channel*. Este tipo de canal de comunicación fue introducido por Elias [20] en 1955. En aquel entonces fue un simple ejemplo teórico, sin embargo, 40 años más tarde, con el desarrollo de Internet, cobró incalculable relevancia.

El *erasure channel* es un canal de comunicación en el cual los símbolos transmitidos, bien son recibidos correctamente, o bien son borrados, debido a que no se han recibido o a que han sido alterados por errores en la transmisión. Es decir, en este tipo de canal sólo pueden ocurrir borrados, no errores; si un símbolo es recibido, el receptor tiene la certeza de que ese símbolo es correcto. Un ejemplo concreto es el *binary erasure channel* en el que los bits son transmitidos a través del canal y la probabilidad de que uno sea borrado es π. Otro caso más general de *erasure channel* es aquel en que el alfabeto $\mathbb{F}_q$ tiene más de 2 elementos, $q > 2$, y cada uno de ellos puede ser no recibido con probabilidad π (véase la figura 2).

Dependiendo del escenario de comunicación, estos vacíos o elementos borrados pueden suceder por diferentes motivos, pero es habitual en canales estructurados como *erasure channels*, que no ocurran aisladamente sino en ráfagas o bloques. Esto quiere decir que la probabilidad de que un elemento sea borrado aumenta si el elemento anterior no ha sido recibido. Por ello, un *erasure channel* puede entenderse como una cadena de Markov de primer orden, donde la probabilidad de que un borrado ocurra precedido de un elemento correctamente recibido es π, mientras que la probabilidad de que éste ocurra después de que un elemento ha sido borrado es θ, donde $\pi < \theta$ (véase la figura 3).

El ejemplo más representativo de *erasure channel* hoy en día es, sin lugar a dudas, Internet. La información se transmite a través de Internet en forma de paquetes, los

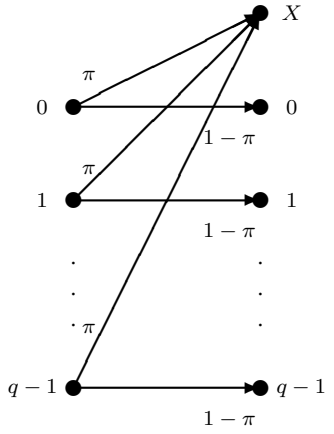


Figura 2: *Canal de borrado sobre alfabetos grandes*

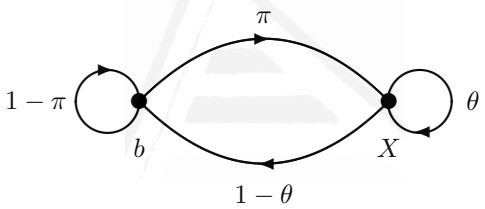


Figura 3: *Representación del canal de borrado como cadena de Markov*

Universitat d'Alacant
Universidad de Alicante

cuales van encabezados por secuencias numéricas que permiten al receptor conocer la posición del paquete en la secuencia de información. De esta forma el receptor conoce la situación de los espacios vacíos en el mensaje. En Internet, el borrado de elementos suele ser provocado por desbordamientos en los buffers cuando reciben un flujo continuo de datos o porque el ruido ha alterado los paquetes. La integridad de los paquetes se suele controlar utilizando códigos de comprobación de redundancia cíclica (códigos CRC).

Dado que los tamaños en bits de los paquetes de información transmitidos a través de Internet son considerables, necesitamos alfabetos con gran número de elementos para representarlos. De esta forma, tomando, por ejemplo, $\mathbb{F} = \mathbb{F}_{2^{1000}}$, si un paquete tiene menos de 1000 bits lo representaremos utilizando el correspondiente elemento en $\mathbb{F}$, y si tiene más de 1000 bits, combinaremos varios elementos para representarlo.

El hecho de que Internet sea el canal de comunicación más usado hoy en día implica el procesamiento de un desbordante volumen de información. Esto dificulta el mantenimiento de la calidad en la comunicación. Los usuarios perciben las saturaciones en la red en forma de respuestas ralentizadas al utilizar las aplicaciones. Esto es debido al protocolo TCP/IP utilizado en Internet. Este protocolo retransmite los paquetes que no han sido recibidos y de esta forma evita las pérdidas de información durante la comunicación. Sin embargo, este procedimiento conlleva un aumento del tiempo de transmisión y ello tiene consecuencias negativas cuando se trata, por ejemplo, de aplicaciones en tiempo real, como son las videollamadas.

Una de las técnicas utilizadas con el propósito de reducir los retrasos en la transmisión es la corrección de errores hacia adelante, conocida en inglés como *forward error correction* (FEC). Este es un mecanismo de corrección de errores que permite su corrección en el receptor sin retransmisión de la información original. Antes de la transmisión, el mensaje es codificado utilizando un cierto código corrector. Las propiedades estructurales del código nos permiten después corregir el mayor número de errores posible.

Hasta ahora los códigos más utilizados para este fin han sido principalmente códigos bloque. De entre ellos, los códigos que alcanzan una mayor capacidad correctora sobre este tipo de canal son los códigos MDS. Un $[N, K]$ código bloque MDS puede corregir hasta $N - K$ símbolos borrados en un bloque de longitud N.

En el capítulo 3 proporcionamos un análisis detallado del comportamiento de los códigos convolucionales sobre *erasure channels*. Utilizando la representación basada en la teoría de módulos introducida en la sección 1.3, consideramos la información recibida como polinomios o secuencias completas a lo largo de las cuales podemos deslizarnos con gran libertad mediante el uso de ventanas deslizantes o corredizas. Esta característica nos permite agrupar la información recibida de diferentes formas y así adaptar nuestro proceso de decodificación a la estructura exacta de la secuencia. Al contrario que en el caso de los códigos bloque, cuya longitud de bloque es fija, el mensaje se puede dividir en ventanas de diferentes tamaños y, además, podemos decidir la posición de las mismas. Gracias a esto, el proceso de decodificación utilizando códigos convolucionales es más flexible y, en muchas ocasiones, se pueden recuperar símbolos no recibidos que un código bloque MDS de la misma longitud no podría recuperar.

Universitat d'Alacant
Universidad de Alicante

En la sección 3.2 demostramos que un (n, k, δ)-código convolucional puede recuperar $d_j^c(\mathcal{C}) - 1$ símbolos borrados en ventanas de longitud $n(j+1)$, donde $d_j^c(\mathcal{C})$ representa la j-ésima distancia columna del código. Además, demostramos que la subclase de códigos convolucionales MDP optimiza este resultado y puede recuperar secuencias completas en tiempo polinómico si en cualquier ventana deslizante de longitud $(L+1)n$ son borrados no más de $(L+1)(n-k)$ símbolos. Aunque esta tasa de recuperación es equivalente a la alcanzada por los códigos bloque MDS, los ejemplos presentados en la sección 3.2 ponen de manifiesto que en muchas situaciones la flexibilidad de los códigos convolucionales permite recuperar información que códigos bloque MDS comparables no pueden.

A pesar de todo, siguen habiendo situaciones en las que gran cantidad de información puede perderse. Con el propósito de retomar el proceso de decodificación cuando ocurren ráfagas de borrados demasiado grandes, introducimos en la sección 3.3 un nuevo tipo de códigos llamados códigos convolucionales MDP-reversos. Estos son códigos que son MDP en ambas direcciones, hacia adelante y hacia atrás, es decir, sus distancias columna y por tanto, sus capacidades correctoras en cada instante, son máximas tanto si avanzamos a lo largo de la secuencia como si retrocedemos. Este hecho nos permite realizar un proceso de recuperación inverso, esto es, retrocediendo a lo largo de la secuencia, y recuperar información que ni un código bloque MDS ni un código convolucional MDP podrían. Demostramos la existencia de estos códigos y, en la sección 3.4, presentamos una construcción concreta, basada en matrices superregulares-reversas, para los casos en que los parámetros (n, k, δ) cumplen que $(n-k) \mid \delta$ y $k > \delta$, o bien, $k \mid \delta$ y $(n-k) > \delta$.

Aunque los códigos convolucionales MDP-reversos son más potentes, el tiempo para reanudar el proceso de decodificación puede ser considerable, ya que necesitamos observar una ventana o parte de la secuencia donde no se produzca ningún borrado. En la sección 3.5 reducimos el tiempo de espera permitiendo, en la reanudación, un cierto número de símbolos borrados por ventana. Para ello introducimos otro nuevo tipo de códigos llamados códigos convolucionales MDP-completos. Los menores de tamaño máximo de la matriz de paridad parcial de estos códigos son no nulos y esto hace que cualquier patrón de borrados en la ventana sea recuperable siempre que no sobrepase el rango de dicha matriz. En otras palabras, ya no tenemos que esperar hasta recibir una ventana completa de símbolos correctamente recibidos, sino simplemente hasta observar una parte de la secuencia donde no ocurran

Universidad de Alicante

muchos borrados. De nuevo, ilustramos con ejemplos el tipo de situaciones que estos códigos son capaces de resolver. Las simulaciones realizadas ratifican que la capacidad correctora de estos códigos supera la de los códigos bloque MDS. Además, y puesto que no hemos encontrado todavía ningún contraejemplo, tendemos a creer que nuestro método de generación de matrices superregulares-reversas junto con la construcción de códigos convolucionales MDP-reversos, propuesta en la sección 3.4, genera de hecho códigos MDP-completos; pero esta conjetura no ha sido probada todavía.

En el capítulo 4 realizamos un estudio similar al anterior, pero en este caso utilizamos la representación entrada-estado-salida de la sección 1.4. En la sección 4.2 probamos de forma alternativa, utilizando la descripción de sistemas lineales, que los códigos convolucionales MDP alcanzan una capacidad correctora máxima sobre el *erasure channel*. Para resolver el problema del tiempo de espera cuando intentamos restablecer el proceso de recuperación, presentamos tres posibles estrategias a seguir dependiendo de las características de nuestro código.

En primer lugar, y haciendo uso de la observabilidad del sistema, establecemos la longitud de la ventana de símbolos correctamente recibidos que debemos observar para poder continuar con el proceso de decodificación. En este caso no sabemos de antemano cuáles son las filas de la matriz de observabilidad que nos proporcionan un sistema con matriz de coeficientes de rango completo.

En segundo lugar, reducimos el tamaño de esa ventana imponiendo más condiciones sobre las matrices. Si la matriz de observabilidad de nuestro código coincide con la matriz de paridad de un código bloque MDS, cualquier combinación de δ filas de la matriz tiene rango δ. De este modo podemos restringir nuestro sistema a las primeras filas de la matriz y conocer *a priori* cuáles serán las ecuaciones de nuestro sistema. En el ejemplo 4.1 mostramos un caso concreto donde las matrices (A, B, C, D) representan un código con estas características.

En último lugar, presentamos una solución para el caso en que no recibimos una ventana completa de símbolos correctamente recibidos, sino que el receptor observa es una ventana con *no demasiados* elementos borrados. El tiempo de espera hasta recibir una ventana donde ningún símbolo haya sido borrado puede ser mucho mayor que el esperado hasta observar una ventana en la que algunos borrados ocurren. Por ello este caso reduce de nuevo la pérdida de información hasta reanudar el

proceso de decodificación. Añadimos condiciones más fuertes sobre el código con el fin de obtener una solución única a nuestro problema. El teorema 4.3 establece que si los subespacios generados por cualquier combinación de $(L+1)(n-k)-\delta$ columnas de la matriz $[-I \mid \mathcal{F}_L]$ son espacios suplementarios del espacio columna generado por la matriz de observabilidad del código $\mathcal{C}$ entonces podemos recalcular el estado del sistema cuando recibimos una ventana con $(L+1)n$ símbolos donde no hay más de $(L+1)(n-k)-\delta$ símbolos borrados. En esta situación se pueden obtener al mismo tiempo los símbolos borrados en esa parte de la secuencia. Este resultado nos permite volver al proceso de recuperación sin necesidad de controlar las posiciones de los borrados. Es decir, ya no es necesario observar un número mínimo de símbolos correctamente recibidos en posiciones contiguas. Proporcionamos un código que satisface las hipótesis del teorema en el ejemplo 4.2.

El capítulo 5 de esta memoria está dedicado a la construcción de códigos convolucionales variantes en el tiempo periódicos estudiados desde el punto de vista de sistemas lineales. La idea de alterar la representación estática de los códigos y transformarla en una descripción variante en el tiempo ha interesado a muchos investigadores y se han obtenido muchos resultados interesantes. Costello [11], proporcionó una cota inferior para la distancia libre de códigos convolucionales no sistemáticos variantes en el tiempo que es mayor que la cota superior para la distancia libre de códigos convolucionales sistemáticos, tanto invariantes como variantes en el tiempo, para cualquier tasa de transmisión. Esto significa que existen códigos convolucionales no sistemáticos variantes en el tiempo que tiene una distancia libre mayor que la de cualquier código convolucional sistemático, variante o invariante, de parámetros comparables. Las grandes implicaciones de este resultado provocaron un especial interés y se profundizó en el estudio de las propiedades particulares y estructuras de estos códigos. Lee [51], constató empíricamente que el número de códigos convolucionales variantes en el tiempo y periódicos aumenta exponencialmente con el periodo para cualquier conjunto de parámetros y además, algunos de ellos, aumentan la distancia libre con respecto a códigos fijos. En [60], Mooser demostró que existen códigos convolucionales variantes periódicos que proporcionan una tasa de error en la decodificación menor que la de cualquier código convolucional fijo de parámetros comparables. También se han estudiado condiciones necesarias y suficientes para asegurar la no catastroficidad del código, así como algoritmos computacionalmente eficientes para comprobarla (véase [8, 61]).

Universitat d'Alacant
Universidad de Alicante

En este capítulo proporcionamos una construcción de códigos convolucionales variantes en el tiempo periódicos utilizando la representación (A, B, C, D) del código, donde las matrices A y D son fijas y sólo B y C varían. En particular, nuestra construcción mantiene el grado o complejidad del código, δ, fijo. Esto evita que la complejidad en el proceso de decodificación aumente. Además, se trata de una generalización de la construcción propuesta por Ogasahara, Kobayashi y Hirasawa en [62]. Los teoremas 5.1 y 5.2 aseguran respectivamente la controlabiliadad y observabilidad (minimalidad y no catastroficidad, respectivamente) de nuestro código variante periódico basándonos en las matrices que representan el código invariante equivalente obtenido. Las pruebas de estos teoremas son constructivas y la construcción de un código variante periódico con estas características se ilustra en el ejemplo 5.1.

Motivados por la existencia de ejemplos en los que la combinación de códigos fijos con distancia libre no óptima proporciona como resultado códigos convolucionales variantes en el tiempo periódicos cuya distancia libre sí lo es, estudiamos en la sección 5.2.2 el comportamiento de las distancias de los códigos obtenidos. El teorema 5.4 proporciona una cota inferior para la distancia libre del código resultante si la matriz de controlabilidad del sistema representa la matriz de paridad de un código bloque MDS. Después estudiamos las distancias columnas del código, las cuales están directamente relacionadas con los códigos convolucionales MDP, parte central en capítulos anteriores. En este caso, aunque las condiciones para que un código convolucional sea MDP son las mismas en el caso variante e invariante, el hecho de que las matrices de nuestros códigos periódicos varíen en el tiempo nos da más flexibilidad y nos permite combinar códigos convolucionales que no son MDP para obtener códigos que sí lo son (véase el ejemplo 5.2). Realizamos el estudio detallado del caso donde los subcódigos que forman el código convolucional variante periódico son $(n, 1, 1)$-códigos convolucionales.

Por último, incluimos las referencias necesarias para el desarrollo de esta tesis, que han sido citadas a lo largo de la misma.

Este prólogo constituye un resumen detallado del contenido de la memoria escrito en una de las dos lenguas oficiales de esta Comunidad Autónoma, siguiendo lo dispuesto en el Acuerdo del Pleno de la Comisión de Doctorado celebrado el día 22 de marzo 2002. El resto de la memoria está escrito en inglés.

Universitat d'Alacant
Universidad de Alicante

Este trabajo ha sido subvencionado por:

- Ayuda para becas y contratos destinada a la formación de doctores concedida por el Vicerrectorado de Investigación, Desarrollo e Innovación de la Universidad de Alicante (UA2007-44773077).
- Proyecto I+D: Construcción de códigos convolucionales. Algoritmos secuenciales y paralelos de decodificación (MTM2005-05759). Subvencionado por el Ministerio de Educación y Ciencia.
- Proyecto I+D: Criptología y seguridad computacional (VIGROB-025). Subvencionado por la Universidad de Alicante (2009, 2010).
- Proyecto I+D: Construcción de funciones bent y códigos LDPC. Aplicaciones criptográficas (ACOMP/2009/142). Subvencionado por la Consellería de Educación.
- Proyecto I+D: Construcción de funciones bent y códigos LDPC. Aplicaciones criptográficas (MTM2008-06674-C02-01). Subvencionado por el Ministerio de Ciencia e Innovación.

Parte de los resultados plasmados en este trabajo aparecen en las actas de las siguientes conferencias, congresos y workshops:

- 4th Workshop on Coding and Systems, Alicante and Elche, Spain, 2008.
- 18th International Symposium on Mathematical Theory of Networks and Systems, Blacksburg, Virginia, USA, 2008.
- First International Workshop on Dynamical Systems and Multidisciplinary Applications, Elche, Spain, 2008.
- Algebra Lineal, Análisis Matricial y Aplicaciones, Vitoria-Gasteiz, España, 2008.
- 18th Symposium on Applied Algebra algorithms and Error Correcting Codes, Tarragona, Spain, 2009.
- 2009 IEEE International Symposium on Information Theory, Seoul, Korea, 2009.
- 5th Workshop on Coding and Systems, Dublin, Ireland, 2009.
- 19th International Symposium on Mathematical Theory of Networks and Systems, Budapest, Hungary, 2010.

Universitat d'Alacant
Universidad de Alicante

Universitat d'Alacant
Universidad de Alicante

Chapter 1

Preliminaries

1.1 Introduction

The transmission of information brings implicit the use of communication channels. Usually these channels are imperfect and suffer the effects of noise and perturbations. This affects the messages sent, therefore, information correctly expressed at the source can arrive altered to the receiver. Coding theory appears as a solution to this problem. Error detecting and correcting codes add redundancy to the information in order to achieve reliable communication (see Figure 1.1). Before being transmitted, the original message is encoded in an optimal way, such that the additional redundancy makes the encoded messages different enough. As a result, the receiver is able to distinguish two different messages when not too many errors occur, that is, the errors can be detected and, in some cases, corrected.

Thanks to the development of error control coding as a mathematical theory we can nowadays enjoy many digital systems, such as computer RAM memories, CDs,

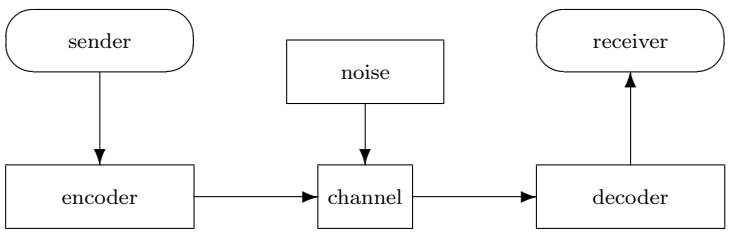


Figure 1.1: *Communication process*

hard disks, wireless communication, digital sound systems, satellite communication, etc.

Error detecting and correcting codes can be divided into two main types: block codes and convolutional codes. Block codes divide the information in blocks of a given length and encode them following always the same procedure. These blocks are encoded independently and the fixed machinery of the code always transforms a certain information into the same encoded message. Block codes have been long studied and its algebraic structure is well known. It is proved that they have good error correcting properties and moreover, algebraic decoding algorithms easy to implement have been given. In Section 1.2 we introduce some preliminaries on block codes.

Convolutional codes consider the information as a whole sequence. Even though they split the information into blocks of a fixed length too, the relative position of each block in the sequence is taken into account. In addition, the blocks are not encoded independently, previous information makes an effect over the next encoded blocks, or in other words, there is overlapping. In this way and oppositely to block codes, a certain information will not always be encoded into the same message since it depends on which is its position within the complete sequence. Although during the last years the algebraic structure of convolutional codes has interested many authors, the knowledge in many aspects is still not as deep as in block code theory.

Convolutional codes have been defined using different points of view. Using linear algebra notions they are studied as linear subspaces of the field of formal Laurent series (see, e.g., [26, 42, 59, 65]). In the literature of symbolic dynamics they are considered as linear, compact, irreducible and shift-invariant subsets of $\mathbb{F}[[z, z^{-1}]]$, where $\mathbb{F}[[z, z^{-1}]]$ represents the ring of formal power series (see, e.g., [48, 53, 57]). And other authors, like Willems (see, e.g., [85, 86, 87]) define them as complete, linear, time-invariant behaviors.

In this chapter we introduce the two descriptions of convolutional codes that we use along the dissertation. The first one is the module-theoretic point of view proposed by Fornasini and Valcher [25, 81] and Rosenthal and some of his coauthors [71, 73, 84]. This definition requires that codewords have finite support and is related to the previously mentioned behaviors by Pontryagin duality (see [70]).

From a systems theory point of view there are several first-order representations

associated to convolutional codes. Kalman [43, 44] showed how the encoding map can be factored resulting in a realization of the generator matrix and later on, Fuhrmann [28] refined this realization procedure. In Section 1.4, we present the systems theory representation that will be relevant for us in this work, the input-state-output representation.

Along the dissertation we consider

- $\mathbb{F}$ a finite field,
- $\mathbf{0}$ a zero vector (of the appropriate size when not indicated),
- O a zero matrix, I the identity matrix (both of them of the appropriate sizes when not indicated),
- $\deg(p(x))$ the degree of a polynomial $p(x)$,
- M' the transpose of a matrix M,
- $\mathrm{colspa}(M)$ the vector space generated by the columns of a matrix M.

1.2 Linear block codes

Definition 1.1: We say that an $[N, K]$ **linear block code** $\mathcal{C}$ over $\mathbb{F}$ is a K-dimensional linear subspace of the vector space $\mathbb{F}^N$. K and N are called the dimension and length of $\mathcal{C}$, respectively. An $N \times K$ matrix G is said to be a **generator matrix** of the code $\mathcal{C}$ if its columns form a basis of $\mathcal{C}$. Then an $[N, K]$ linear block code $\mathcal{C}$ is described as

$$\mathcal{C} = \{G\boldsymbol{u} \in \mathbb{F}^N \mid \boldsymbol{u} \in \mathbb{F}^K\}.$$

The **information vector** $\boldsymbol{u}$ of length K is encoded into the **code vector** or **codeword** $\boldsymbol{v}$ of length N, then we say that the code has **rate** $R = K/N$.

Note that the fact that G is a constant scalar matrix makes us think about the encoding process as an invariable mechanism that transforms the information vectors into codewords using always the same algorithm. This perception is one of the main differences that one can find between block codes and convolutional codes (see Section 1.3). Another important distinction is the idea that in block coding any information vector is only related to the corresponding code vector generated at that instant. We show in the next section that this is not the case for convolutional

codes.

Another way to describe this linear subspace is through a kernel representation. Indeed, there exists an $(N-K)\times N$ matrix H called a **parity check matrix** of $\mathcal{C}$, such that $\boldsymbol{v}\in\mathcal{C}$ if and only if $H\boldsymbol{v}=\mathbf{0}$. In general, G and H are not unique due to the fact that one can write many different bases for a subspace (see, e.g., [56]).

One of the most important concepts of a block code is the **minimum distance**, denoted by $d_{\min}(\mathcal{C})$. If $\boldsymbol{v}\in\mathbb{F}^N$ the **Hamming weight** of $\boldsymbol{v}$, $\mathrm{wt}(\boldsymbol{v})$, is defined as the number of nonzero components of $\boldsymbol{v}$, then the minimum distance of $\mathcal{C}$ is given by the following expression

$$d_{\min}(\mathcal{C})=\min\{\mathrm{wt}(\boldsymbol{v})\mid \boldsymbol{v}\neq\mathbf{0}\quad\text{with}\quad \boldsymbol{v}\in\mathcal{C}\}.$$

One way of computing this parameter is using the parity check matrix of the code. If any combination of s columns of H is linearly independent but there exist $s+1$ columns that are linearly dependent, then $d_{\min}(\mathcal{C})=s+1$ (see, e.g., [68]).

The minimum distance is directly related to the error correction and detection capabilities of the code. If $d_{\min}(\mathcal{C})=d$ then the code can detect at most $d-1$ errors occurring during the transmission of a codeword and can correct at most $\left\lfloor\frac{d-1}{2}\right\rfloor$ of them (see, e.g., [69]). Therefore, in order to achieve reliable communication, is important to focus on the construction of codes with large minimum distance. However, $d_{\min}(\mathcal{C})$ cannot attain any value since we have the following upper bound.

Theorem 1.1 (Singleton bound): *If $\mathcal{C}$ is an $[N,K]$ linear block code then $d_{\min}(\mathcal{C})$ satisfies*

$$d_{\min}(\mathcal{C})\leq N-K+1. \tag{1.1}$$

*If $d_{\min}(\mathcal{C})=N-K+1$, then $\mathcal{C}$ is said to be **maximum distance separable** (MDS).*

MDS block codes have the capacity of correcting the maximum number of errors among all the block codes of rate K/N. Further details and examples of these types of codes can be found in the literature (see, e.g., [35, 52, 68] and the references therein).

1.3 Convolutional codes: module point of view

In this section we introduce the central concept of this dissertation, the concept of convolutional code. We present in detail the main results and properties needed for the rest of the discussion. The approach followed is that of module theory and it will be the one used in Chapter 3.

The previous section was dedicated to block codes. We saw that those can be thought as black boxes where the information goes in, is transformed, and then is sent out, and this process is invariably made in the same way, that is, the same information will always be encoded into the same codeword. This fact suggests the question of what happens if this process varies, in other words, if time matters. If we introduce a variable z, usually called the delay operator, to indicate the instant in which each information arrived or each codeword was transmitted, then we can represent the message as a polynomial sequence

$$\mathbf{u}(z) = \boldsymbol{u}_0 + \boldsymbol{u}_1 z + \cdots + \boldsymbol{u}_l z^l \in \mathbb{F}^k[z]$$

and the codeword in a similar way

$$\mathbf{v}(z) = \boldsymbol{v}_0 + \boldsymbol{v}_1 z + \cdots + \boldsymbol{v}_l z^l \in \mathbb{F}^n[z].$$

A block code is now described as the following set

$$\mathcal{C} = \left\{ G\mathbf{u}(z) \mid \mathbf{u}(z) \in \mathbb{F}^k[z] \right\} = \mathrm{im}_{\mathbb{F}[z]} G.$$

In this representation the encoding process is still static because the generator matrix G is fixed. To make possible that a certain input is not encoded as the same codeword at two different instants we allow the generator matrix to be a polynomial matrix instead of a scalar matrix,

$$G(z) = G_0 + G_1 z + \cdots + G_m z^m.$$

Then we obtain the notion of convolutional code (see [32, 70, 71]).

Universitat d'Alacant
Universidad de Alicante

Definition 1.2: A **convolutional code** $\mathcal{C}$ of rate k/n is a submodule of $\mathbb{F}^n[z]$ that can be described as

$$\mathcal{C} = \left\{ \mathbf{v}(z) \in \mathbb{F}^n[z] \mid \mathbf{v}(z) = G(z)\mathbf{u}(z) \text{ with } \mathbf{u}(z) \in \mathbb{F}[z]^k \right\}$$

where $G(z)$ is an $n \times k$ full rank polynomial matrix called a **generator matrix** of $\mathcal{C}$, $\mathbf{u}(z)$ is the **information vector** and $\mathbf{v}(z)$ is the **code vector** or **codeword**. m is called the **memory** of the generator matrix.

The idea is that a convolutional code *remembers* past inputs and those affect to the new messages bringing variations on the encoding process. The memory specifies for how long the information has influence on the outputs.

Two generator matrices $G_1(z)$ and $G_2(z)$ generate the same convolutional code $\mathcal{C}$ if there exists a unimodular matrix $U(z) \in \mathbb{F}^{k\times k}[z]$, that is, the determinant of $U(z)$ is a nonzero element of $\mathbb{F}$, such that $G_1(z) = G_2(z)U(z)$.

An important invariant of a convolutional code is the **degree** or **complexity** of $\mathcal{C}$, denoted by δ and defined as the maximum of the degrees of the determinants of the $k \times k$ submatrices of any generator matrix of $\mathcal{C}$. Since the equivalence relation between generator matrices presented above preserves the degrees of these determinants, this definition makes sense. Then we say that $\mathcal{C}$ is an (n, k, δ)-convolutional code [59].

Universitat d'Alacant
Universidad de Alicante

If $G(z) = [g_{ij}(z)]$ then the **column degrees** of the generator matrix are defined as

$$\delta_j = \max\{\deg(g_{ij}(z)),\ i = 1, 2, \ldots, n\}, \quad \text{for} \quad j = 1, 2, \ldots, k.$$

The index δ_j is also called the **constraint length for the j-th input** of the matrix $G(z)$ (see [42]).

The **high-order coefficients matrix** of $G(z)$, G_∞, is the matrix whose j-th column is formed by the coefficients of z^{δ_j} in the j-th column of $G(z)$. If G_∞ has full rank, then $G(z)$ is called a **minimal generator matrix**. If $G(z)$ is minimal the column degrees $\{\delta_j\}_{j=0}^k$ are called the **Forney indices** of the code (see, e.g., [59]) and it holds that $\delta = \sum_{j=1}^k \delta_j$. The matrix is called minimal because δ attains in this case the minimal possible value. Note that the Forney indices of a code are unique; if $G_1(z)$ and $G_2(z)$ are two minimal generator matrices of $\mathcal{C}$ then both have

the same column degrees (see, e.g., [90]).

We say that a code $\mathcal{C}$ is **observable** (see, e.g., [70, 73]) if the generator matrix $G(z)$ is right-prime, that is, if its $k \times k$ minors are nonzero and they only have nontrivial common factors (z^r, with $r \in \mathbb{N}$, is considered a trivial factor). This avoids the type of so-called **catastrophic** situations in which an information vector $\mathbf{u}(z)$ with an infinite number of nonzero coefficients can be encoded into a code vector $\mathbf{v}(z)$ with an infinite number of zero coefficients. If this would be the case finitely many errors on the received vector could translate into infinitely many errors when recovering the original information. Therefore, only observable codes are regularly considered. We assume this condition on the codes appearing in the rest of the dissertation.

If $\mathcal{C}$ is an observable code then it can be equivalently described as the right $\mathbb{F}[z]$-kernel of an $(n-k) \times n$ full rank polynomial matrix $H(z)$. $\mathcal{C}$ is now described as the set

$$\mathcal{C} = \{\mathbf{v}(z) \in \mathbb{F}^n[z] \;\mid\; H(z)\mathbf{v}(z) = \mathbf{0}\}\,.$$

where $H(z)$ is called a **parity check matrix** of the code. If we rewrite $H(z)$ as a matrix polynomial

$$H(z) = H_0 + H_1 z + \cdots + H_\nu z^\nu$$

where $H_i = O$ for $i > \nu$, we can expand the kernel representation in the following way

$$\begin{bmatrix} H_0 & & & & & \\ \vdots & \ddots & & & & \\ H_\nu & \cdots & H_0 & & & \\ & \ddots & & \ddots & & \\ & & H_\nu & \cdots & H_0 & \\ & & & \ddots & \vdots & \\ & & & & H_\nu & \end{bmatrix} \begin{bmatrix} \boldsymbol{v}_0 \\ \boldsymbol{v}_1 \\ \vdots \\ \boldsymbol{v}_l \end{bmatrix} = \mathbf{0}. \tag{1.2}$$

where the matrix has size $(l+\nu+1)(n-k) \times (l+1)n$. Equation (1.2) will be relevant in Chapter 3.

Identically to the case of block codes, distance measures are of main importance. In order to define the distance concepts that concern us in our discussion, we first

Universitat d'Alacant
Universidad de Alicante

need to define the **weight** of a code vector $\mathbf{v}(z)$. This is denoted by $\mathrm{wt}(\mathbf{v}(z))$ and is the sum of the Hamming weights of the coefficients of $\mathbf{v}(z)$, that is, $\mathrm{wt}(\mathbf{v}(z)) = \sum_{i=0}^{l} \mathrm{wt}(\boldsymbol{v}_i)$.

We define the **free distance**, $d_{\mathrm{free}}(\mathcal{C})$, of a convolutional code as

$$d_{\mathrm{free}}(\mathcal{C}) = \min\left\{\mathrm{wt}(\mathbf{v}(z)) \;\mid\; \mathbf{v}(z) \in \mathcal{C} \quad \text{and} \quad \mathbf{v}(z) \neq 0\right\}.$$

This is the analogue of minimum distance in the case of block codes.

Rosenthal and Smarandache [72] showed that the free distance of an (n, k, δ)-convolutional code is upper bounded by

$$d_{\mathrm{free}}(\mathcal{C}) \leq (n-k)\left(\left\lfloor \frac{\delta}{k} \right\rfloor + 1\right) + \delta + 1. \tag{1.3}$$

This bound is known as the **generalized Singleton bound** since it generalizes in a natural way the Singleton bound for block codes. If we let $\delta = 0$, that is, we assume $\mathcal{C}$ is a block code, then we obtain the expression (1.1). Moreover, an (n, k, δ)-convolutional code is defined to be a **maximum distance separable convolutional code** (MDP) if its free distance achieves the generalized Singleton bound.

A more local distance measure, but the crucial one in big part of this thesis, is the **j-th column distance** of $\mathcal{C}$ (see [41]) given by the expression

$$d_j^c(\mathcal{C}) = \min\left\{\mathrm{wt}(\mathbf{v}_{[0,j]}(z)) \mid \mathbf{v}(z) \in \mathcal{C} \quad \text{and} \quad \boldsymbol{v}_0 \neq \mathbf{0}\right\}$$

where $\mathbf{v}_{[0,j]}(z) = \boldsymbol{v}_0 + \boldsymbol{v}_1 z + \cdots + \boldsymbol{v}_j z^j$ represents the j-th truncation of the code vector $\mathbf{v}(z) \in \mathcal{C}$. It is related with the $d_{\mathrm{free}}(\mathcal{C})$ in the following way

$$d_{\mathrm{free}}(\mathcal{C}) = \lim_{j \to \infty} d_j^c(\mathcal{C}). \tag{1.4}$$

If we define the matrix

$$\mathcal{H}_j = \begin{bmatrix} H_0 & & & \\ H_1 & H_0 & & \\ \vdots & \vdots & \ddots & \\ H_j & H_{j-1} & \cdots & H_0 \end{bmatrix} \tag{1.5}$$

then the following theorem characterizes the column distances of a convolutional code.

Universitat d'Alacant Universidad de Alicante

Theorem 1.2 ([33], Proposition 2.1): *Let $d \in \mathbb{N}$. The following properties are equivalent.*

(a) $d_j^c = d$;

(b) *none of the first n columns of $\mathcal{H}_j$ is contained in the span of any other $d-2$ columns and one of the first n columns of $\mathcal{H}_j$ is in the span of some other $d-1$ columns of that matrix.*

The j-th column distance is upper bounded,

$$d_j^c(\mathcal{C}) \leq (n-k)(j+1)+1, \tag{1.6}$$

and the maximality of any of the column distances implies the maximality of all the preceding ones [33], that is, if $d_j^c(\mathcal{C}) = (n-k)(j+1)+1$ for some j, then

$$d_i^c(\mathcal{C}) = (n-k)(i+1)+1 \quad \text{for all} \quad i \leq j.$$

The $(m+1)$-tuple $(d_0^c(\mathcal{C}), d_1^c(\mathcal{C}), \ldots, d_m^c(\mathcal{C}))$ is called the **column distance profile** of the code [42]. Since no column distance can achieve a value greater than the generalized Singleton bound, the largest integer j for which that bound (1.6) can be attained is for $j = L$, with

$$L = \left\lfloor \frac{\delta}{k} \right\rfloor + \left\lfloor \frac{\delta}{n-k} \right\rfloor. \tag{1.7}$$

We define now a subclass of convolutional codes that holds our attention along the thesis. An (n, k, δ)-convolutional code $\mathcal{C}$ is said to have a **maximum distance profile** (MDP) (see, [33, 39]) if $d_L^c(\mathcal{C}) = (n-k)(L+1)+1$. $d_L^c(\mathcal{C})$ is maximal, therefore all the previous column distances attain the bound (1.6). One can say that the column distances of an MDP code are maximal until is no longer possible. These codes are very interesting since its maximum possible growth in the column distances means that they have the potential to correct a maximal number of errors per time interval. MDP convolutional codes are similar to MDS block codes within windows of size $(j+1)n$. Therefore we consider MDP codes a more appropriated analogue of MDS block codes than MDS convolutional codes.

Universitat d'Alacant
Universidad de Alicante

Let us define the matrix

$$\mathcal{G}_j = \begin{bmatrix} G_0 & & & \\ G_1 & G_0 & & \\ \vdots & \vdots & \ddots & \\ G_j & G_{j-1} & \cdots & G_0 \end{bmatrix}'. \tag{1.8}$$

The following result characterizes the maximality of the column distances of $\mathcal{C}$ and hence, for $j = L$, provides us with an algebraic description of MDP convolutional codes.

Theorem 1.3 ([33], Theorem 2.4): *Let $\mathcal{C}$ be an (n, k, δ)-convolutional code. Let $\mathcal{G}_j$ and $\mathcal{H}_j$ be like in (1.8) and (1.5). Then the following are equivalent:*

(a) $d_j^c(\mathcal{C}) = (n-k)(j+1)+1$;

(b) *every $(j+1)k \times (j+1)k$ full-size minor of $\mathcal{G}_j$ formed from the columns with indices $1 \leq t_1 < \cdots < t_{(j+1)k}$, where $t_{sk+1} > sn$ for $s = 1, 2, \ldots, j$, is nonzero;*

(c) *every $(j+1)(n-k)\times(j+1)(n-k)$ full-size minor of $\mathcal{H}_j$ formed from the columns with indices $1 \leq r_1 < \cdots < r_{(j+1)(n-k)}$, where $r_{s(n-k)} \leq sn$ for $s = 1, 2, \ldots, j$, is nonzero.*

In particular, when $j = L$, $\mathcal{C}$ is an MDP convolutional code.

Universitat d'Alacant
Universidad de Alicante

Note that the index conditions in parts (b) and (c) mean that the minors under consideration are those which have the possibility to be nonzero.

1.4 Convolutional codes: systems theory point of view

In this section we review an additional representation of convolutional codes necessary in Chapters 4 and 5. We consider a linear systems theory description known as the input-state-output representation. This description was adopted by Massey and Sain [58]. In order to introduce it we show the connection between different first order representations.

As presented in [73] we have the following existence theorem.

Theorem 1.4: *Assume $\mathcal{C} \subset \mathbb{F}^n[z]$ is a rate k/n convolutional code of degree δ. Then there exist matrices $K, L \in \mathbb{F}^{(\delta+n-k)\times\delta}$ and $M \in \mathbb{F}^{(\delta+n-k)\times n}$ such that:*

$$\mathcal{C} = \{\mathbf{v}(z) \in \mathbb{F}^n[z] \mid (zK + L)\mathbf{x}(z) + M\mathbf{v}(z) = \mathbf{0} \text{ for some } \mathbf{x}(z) \in \mathbb{F}^\delta[z]\}.$$

Moreover, the following conditions are satisfied:

(a) *K has full column rank;*
(b) *$[K \mid M]$ has full row rank;*
(c) *$\operatorname{rank}[z_0K + L \mid M] = \delta + n - k$, for all $z_0 \in \mathbb{K}$, where $\mathbb{K}$ is the algebraic closure of $\mathbb{F}$.*

If we perform the suitable similarity transformations and permutations of the components of $\mathbf{v}(z)$ we may rewrite the (K, L, M) triple in the following way

$$K = \begin{bmatrix} I_\delta \\ O \end{bmatrix}, \quad L = \begin{bmatrix} -A \\ -C \end{bmatrix}, \quad M = \begin{bmatrix} O & -B \\ I_{n-k} & -D \end{bmatrix},$$

where $A \in \mathbb{F}^{\delta\times\delta}$, $B \in \mathbb{F}^{\delta\times k}$, $C \in \mathbb{F}^{(n-k)\times\delta}$ and $D \in \mathbb{F}^{(n-k)\times k}$. If we partition the vector $\boldsymbol{v}_t$ into $\boldsymbol{v}_t = \begin{bmatrix} \boldsymbol{y}_t \\ \boldsymbol{u}_t \end{bmatrix}$, where $\boldsymbol{y}_t$ has $n-k$ components and $\boldsymbol{u}_t$ has k components, then the convolutional code is equivalently described by the familiar-looking (A, B, C, D) representation

$$\left.\begin{array}{rcl} \boldsymbol{x}_{t+1} &=& A\boldsymbol{x}_t + B\boldsymbol{u}_t \\ \boldsymbol{y}_t &=& C\boldsymbol{x}_t + D\boldsymbol{u}_t \end{array}\right\}, \quad t = 0, 1, 2, \ldots, \quad \boldsymbol{x}_0 = \mathbf{0}. \tag{1.9}$$

This system is known as the **input-state-output representation** of $\mathcal{C}$. We call $\boldsymbol{x}_t \in \mathbb{F}^\delta$ the **state vector** of the system and $\boldsymbol{y}_t \in \mathbb{F}^{n-k}$ the **parity vector**. The encoding that results from the equations in (1.9) is a so-called **systematic encoding**; this means that the information vectors appear in the code vectors (see, e.g., [71, 73, 90] for more details).

Universitat d'Alacant
Universidad de Alicante

Let j be a positive integer, then we define the matrices

$$\Phi_j(A,B) = \begin{bmatrix} B & AB & \cdots & A^{j-2}B & A^{j-1}B \end{bmatrix} \quad \text{and} \quad \Omega_j(A,C) = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{j-2} \\ CA^{j-1} \end{bmatrix}. \quad (1.10)$$

They are called the **controllability** and **observability matrices** of the system, respectively. In that case we say that (A,B) is a **controllable** pair if $\operatorname{rank}\Phi_\delta(A,B) = \delta$. Then the smallest integer κ such that $\operatorname{rank}\Phi_\kappa(A,B) = \delta$ is called the controllability index of (A,B). In a similar manner, (A,C) is called an **observable** pair if $\operatorname{rank}\Omega_\delta(A,C) = \delta$ and the smallest integer μ such that $\operatorname{rank}\Omega_\mu(A,C) = \delta$ is called the observability index of (A,C).

If (A,B) is a controllable pair, it is possible to drive a given state vector to any other state vector in a finite number of steps. If (A,C) is an observable pair, then it is possible to calculate the state vector of the system at time t_0 by observing a finite number of outputs starting with t_0. This notion will be important later on in this work.

If (A,B) is a controllable pair and (A,C) is an observable pair, then the input-state-output representation (A,B,C,D) is called a **minimal realization**, δ is called the **McMillan degree** of the system and it attains its minimum value. For simplicity, we assume that (A,B) forms a controllable pair and (A,C) forms an observable pair.

We can now define the already mentioned distance measures from a systems theory point of view. The free distance is expressed now as

$$d_{\text{free}}(\mathcal{C}) = \min\left\{\sum_{t=0}^{\infty}\operatorname{wt}(\boldsymbol{y}_t) + \sum_{t=0}^{\infty}\operatorname{wt}(\boldsymbol{u}_t)\right\},$$

where the minimum is taken over all possible nonzero codewords. Truncating the codewords at the j-th iteration of the system we obtain the expression for the j-th column distance

$$d_j^c(\mathcal{C}) = \min_{\boldsymbol{u}_0\neq\boldsymbol{0}}\left\{\sum_{t=0}^{j}\operatorname{wt}(\boldsymbol{y}_t) + \sum_{t=0}^{j}\operatorname{wt}(\boldsymbol{u}_t)\right\}.$$

Universitat d'Alacant
Universidad de Alicante

In Chapter 4 we focus our attention again on the subclass of MDP convolutional codes and we use then the (A, B, C, D) representation. The algebraic characterization of MDP codes using linear systems theory can be found in [37, 39] and it is developed in the following way. Let $m \in \{1, 2, \ldots, \min\{\gamma(n-k), \gamma k\}\}$ and let $I = \{i_1, \ldots, i_m\}$, with $i_1 < \cdots < i_m$ be a set of row indices, and $J = \{j_1, \ldots, j_m\}$, with $j_1 < \cdots < j_m$ be a set of column indices. We denote by A_J^I the $m \times m$ submatrix of a matrix A formed by intersecting the rows indexed by the members of I with the columns indexed by the members of J. Then we have the following definition.

Definition 1.3 ([37], Definition 1.3): Consider a lower triangular block Toeplitz matrix

$$\mathcal{T} = \begin{bmatrix} T_0 & 0 & \cdots & 0 \\ T_1 & T_0 & \ddots & \vdots \\ \vdots & & \ddots & 0 \\ T_\gamma & \cdots & T_1 & T_0 \end{bmatrix} \in \mathbb{F}^{\gamma(n-k)\times\gamma k},$$

where each block has size $(n-k) \times k$. Let $I = \{i_1, \ldots, i_m\}$ be a set of row indices and $J = \{j_1, \ldots, j_m\}$ a set of column indices of $\mathcal{T}$. $\mathcal{T}_J^I$ is said to be **proper** if, for each $\sigma \in \{1, 2, \ldots, m\}$, the inequality $j_\sigma \leq \left\lceil \frac{i_\sigma}{n-k} \right\rceil k$ holds.

Universitat d'Alacant
Universidad de Alicante

The minors related to proper submatrices are called **not trivially zero** minors. The rest of minors are said to be trivially zero. If $\mathcal{T}$ is a lower triangular matrix, that is, if $n = 2$ and $k = 1$, we say that $\mathcal{T}$ is **superregular** if every proper submatrix has a nonzero determinant. A deeper study of this notion will be developed in Chapter 2.

Iterating the equations defining system (1.9) we obtain the following relation

$$\begin{bmatrix} \boldsymbol{y}_t \\ \boldsymbol{y}_{t+1} \\ \vdots \\ \boldsymbol{y}_{\gamma-1} \\ \boldsymbol{y}_\gamma \end{bmatrix} = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{\gamma-t-1} \\ CA^{\gamma-t} \end{bmatrix} \boldsymbol{x}_t$$

$$+\begin{bmatrix} D & & & & \\ CB & D & & & \\ \vdots & & \ddots & & \\ CA^{\gamma-t-2} & CA^{\gamma-t-3} & \cdots & D & \\ CA^{\gamma-t-1}B & CA^{\gamma-t-2}B & \cdots & CB & D \end{bmatrix} \begin{bmatrix} \boldsymbol{u}_t \\ \boldsymbol{u}_{t+1} \\ \vdots \\ \boldsymbol{u}_{\gamma-1} \\ \boldsymbol{u}_{\gamma} \end{bmatrix}. \tag{1.11}$$

This expression is known as the **local description of trajectories** and together with the evolution of the state

$$\boldsymbol{x}_\lambda = A^{\lambda-t}\boldsymbol{x}_t + \begin{bmatrix} A^{\lambda-t-1}B & \cdots & AB & B \end{bmatrix} \begin{bmatrix} \boldsymbol{u}_t \\ \vdots \\ \boldsymbol{u}_{\lambda-1} \end{bmatrix}, \ \lambda = t+1, t+2, \ldots \gamma+1, \tag{1.12}$$

describes the behavior of the system starting at any instant t.

The block Toepliz matrix appearing in equation (1.11) is important in the rest of the discussion. We denote it by

$$\mathcal{F}_j = \begin{bmatrix} F_0 & & & \\ F_1 & F_0 & & \\ \vdots & & \ddots & \\ F_j & F_{j-1} & \cdots & F_0 \end{bmatrix} = \begin{bmatrix} D & & & \\ CB & D & & \\ \vdots & & \ddots & \\ CA^{j-1}B & CA^{j-2}B & \cdots & D \end{bmatrix}. \tag{1.13}$$

The following theorem establishes the relation between the maximality of the column distances of $\mathcal{C}$ and the matrix in (1.13). This is a first step toward the characterization of MDP convolutional codes.

Theorem 1.5 ([39], Theorem 2.4): *Let $\mathcal{C}$ be an (n, k, δ)-convolutional code described by matrices (A, B, C, D) and consider the block Toepliz matrix $\mathcal{F}_j$ introduced in (1.13). Then $\mathcal{C}$ has j-th column distance $d_j^c(\mathcal{C}) = (n-k)(j+1)+1$ if and only if there are no zero entries in F_i, $0 \leq i \leq j$, and every minor of $\mathcal{F}_j$ which is not trivially zero is nonzero.*

Finally, making use of the fact that the maximality of $d_L^c(\mathcal{C})$ implies the maximality of all the column distances, we have an algebraic criterion for a convolutional code to be MDP (see, e.g., [33, 39]).

Universitat d'Alacant
Universidad de Alicante

Corollary 1.1 ([39], Corollary 2.5): *Let* $L = \left\lfloor \frac{\delta}{k} \right\rfloor + \left\lfloor \frac{\delta}{n-k} \right\rfloor$. *Then, the matrices* (A, B, C, D) *generate an MDP* (n, k, δ)*-convolutional code, if and only if, the matrix* $\mathcal{F}_L$ *has the property that every minor which is not trivially zero is nonzero.*

Note that Corollary 1.1 studies the nonsigularity of the not trivially zero minors of all sizes in $\mathcal{F}_L$. However, the result presented in the previous section, Theorem 1.3, referred only to the full size minors of matrices $\mathcal{G}_L$ and $\mathcal{H}_L$ in equations (1.8) and (1.5). A similar setting can be obtained making use of the following description. Taking into account that $\boldsymbol{x}_0 = \mathbf{0}$ and making some transformations on equation (1.11) we obtain

$$\left[\begin{array}{c|cccc} & D & & & \\ & CB & D & & \\ -I & CAB & CB & D & \\ & \vdots & \vdots & & \ddots \\ & CA^{L-1}B & CA^{L-2}B & \cdots & CB \quad D \end{array}\right] \left[\begin{array}{c} \boldsymbol{y}_0 \\ \boldsymbol{y}_1 \\ \vdots \\ \boldsymbol{y}_L \\ \hline \boldsymbol{u}_0 \\ \boldsymbol{u}_1 \\ \vdots \\ \boldsymbol{u}_L \end{array}\right] = \mathbf{0}. \tag{1.14}$$

Then Corollary 1.1 translates into the property that $\mathcal{C}$ is an MDP convolutional code if and only if each of the $(L+1)(n-k) \times (L+1)(n-k)$ not trivially zero full size minors of matrix $[-I \mid \mathcal{F}_L]$ in (1.14) is nonzero.

Universitat d'Alacant
Universidad de Alicante

Universitat d'Alacant
Universidad de Alicante

Chapter 2

Generation of superregular and reverse-superregular matrices

2.1 Introduction

Edrei [17, 18] and Aissen et al. [2] introduced the concept of totally positive sequences. These are infinite real sequences $\{a_n\}$ such that its corresponding 4-way infinite matrix

$$\begin{bmatrix} & \vdots & \vdots & \vdots & \\ \cdots & a_0 & a_{-1} & a_{-2} & \cdots \\ \cdots & a_1 & a_0 & a_{-1} & \cdots \\ \cdots & a_2 & a_1 & a_0 & \cdots \\ & \vdots & \vdots & \vdots & \end{bmatrix}$$

has all the minors of all orders non-negative. These matrices are called totally positive matrices (totally nonnegative for some authors [22, 49]).

It was proved [19] that the functions that generate totally positive sequences such that $a_n = 0$ for $n < 0$ and $a_0 = 1$ providing with the 2-way infinite matrix

$$\begin{bmatrix} 1 & 0 & 0 & 0 & \cdots \\ a_1 & 1 & 0 & 0 & \cdots \\ a_2 & a_1 & 1 & 0 & \cdots \\ a_3 & a_2 & a_1 & 1 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \end{bmatrix} \tag{2.1}$$

are of the form

$$g(z) = 1 + a_1 z + a_2 z^2 + a_3 z^3 + \cdots = \frac{\prod_{i=1}^{\infty}(1+\epsilon_i z)}{\prod_{i=1}^{\infty}(1-\eta_i z)} \exp(\gamma z) \tag{2.2}$$

with $\gamma \geq 0$, $\epsilon_i \geq 0$, $\eta_i \geq 0$, $\sum_{i=0}^{\infty} \epsilon_i < \infty$ and $\sum_{i=0}^{\infty} \eta_i < \infty$.

Totally positive matrices appear in contexts like control sets [7], stochastic processes and approximation theory [30, 45], coding theory [37, 74], convexity [79], distribution of zeros of entire functions [13] and its geometric theory point of view [82], oscillation in mechanical systems [29], planar resistor networks [14] or Pólya frequency sequences [75]. (For further references see, e.g., [3, 45]). Sufficient conditions and efficient ways to test total positivity have been considered as well (see, e.g., [12, 24, 46]).

The matrices that hold our attention in this chapter are the so-called **superregular matrices**. They can be considered as the correspondent of totally positive matrices for finite fields since they are finite totally positive matrices with elements over a finite field $\mathbb{F}$ whose minors having the possibility to be nonzero, are nonzero. Recalling the notion of proper submatrix mentioned in Definition 1.3 we have the following definition.

Universitat d'Alacant
Universidad de Alicante

Definition 2.1: Let A be an $l \times l$ lower triangular matrix

$$A = \begin{bmatrix} a_0 & 0 & \cdots & 0 & 0 \\ a_1 & a_0 & \cdots & 0 & 0 \\ \vdots & \vdots & & \vdots & \vdots \\ a_{l-1} & a_{l-2} & \cdots & a_0 & 0 \\ a_l & a_{l-1} & \cdots & a_1 & a_0 \end{bmatrix}$$

where $a_i \in \mathbb{F}$, $i = 0, 1, \ldots, l$. A is said to be superregular if every proper submatrix of A has a nonzero determinant.

From now on we will refer to the finite sequence $a_0 + a_1 z + \cdots + a_l z^l$ associated

to the matrix A as the *generator polynomial* of the matrix.

Superregular matrices with elements over finite fields appear in the context of coding theory. They are of special interest when constructing MDS block codes [74] or MDP convolutional codes (see definitions in Chapter 1). In [39], the exact relation between this type of matrices and the algebraic characterization of MDP convolutional codes is given and later on a concrete construction of MDP codes using superregular matrices is presented [37].

The minimum field sizes required for the existence of these matrices have been obtained using computer algebra programs and upper bounds on the required field size have been proved in [37]. Some actions preserving superregularity were presented as well. In [33], a construction for $n \times n$ superregular matrices over $\mathbb{F}_p$ is given. However, p must be considered large enough and this results in a field size much larger than the minimum required. It is not clear how one can find a good bound about how large p must be for a given n. It was conjectured that for every $l \geq 5$ one can find a superregular $(l \times l)$-Toeplitz matrix over $\mathbb{F}_{2^{l-2}}$, but this is still an open question and a construction has never been given. Very little is known about how to generate this kind of matrices.

In this chapter, we introduce a construction of superregular matrices which allows us to generate these matrices over fields of the form $\mathbb{F} = \mathbb{F}_{p^n}$ and we provide actions that preserve this property. The fact of considering these kind of matrices as the corresponding objects of the totally positive ones over finite fields produces very powerful tools to develop a straight forward generating method. We compute the generator polynomial of the matrix as a product $\prod_{i=1}^{l-1}(1+\epsilon_i z)$, where the value of ϵ_i depends on the choice of a generator element $\alpha \in \mathbb{F}$, or an irreducible polynomial in the case $\mathbb{F} = \mathbb{F}_{p^n}$. Although we cannot *a priori* decide which is the irreducible polynomial over $\mathbb{F}_{p^n}$ or the generator element over $\mathbb{F}_p$ that one must choose in order to construct the generator polynomial of the matrix, our method reduces drastically the exhaustive computer searches, since the number of irreducible polynomials or generator elements in a field is much smaller than the number of combinations of elements one should check. Thus, the method given in Section 2.2.1 can be considered as the first direct one that generates superregular matrices over $\mathbb{F}_{p^n}$.

Procedures used to test efficiently the total-positivity of a matrix are not valid

in our case since many of them are based on decompositions that use the Cauchy-Binet formula for the determinant of the product of two nonsquare matrices, that is, $\det(AB) = \sum_S \det(A^S)\det(B_S)$ where A is a $k \times n$ matrix, B is a $n \times k$ matrix, $S = (s_1, s_2, \ldots, s_k)$, $1 \leq s_1 < s_2 < \cdots < s_k \leq n$, runs through all such multi-indices, A^S denotes the matrix formed from A using the columns with indices in S in that order and B_S denotes the matrix formed from B using the rows with indices in S in that order. Since the arithmetic given over the real numbers does not hold the same properties as the arithmetic over finite fields, such as that the sum of two nonzero elements is different from zero, these tests are not useful in our situation. Due to this reason it is an important task to find mechanisms to reduce the size and effort of computer searches and superregularity tests.

In Section 2.3, we prove that this construction produces matrices that have twice the superregular property, in a forward and backward sense. We call them **reverse-superregular matrices**. The particular properties of these matrices have already been used in the construction of special codes, such as reverse-MDP convolutional codes (see Chapter 3), and suggest the idea that they could apply for obtaining other codes with good characteristics.

To conclude, in Section 2.4 we compare the minimum field sizes required for superregularity and reverse-superregularity and show that our construction approximates better to this minimum when compared with the construction given in [33]. In addition, we give lists of irreducible polynomials and generator elements that provide with superregular matrices when using our method.

2.2 Construction of superregular matrices

In this section a method to generate superregular matrices is presented. The motivation for this construction is the idea of considering superregular matrices as the corresponding objects over finite fields of totally positive matrices. We base our construction on the same kind of factors that generate totally positive sequences (see expression (2.2)). From the three type of factors, the one of the form $\exp(\gamma z)$ cannot be translated into the context of finite fields and therefore we do not consider it for our construction. Hence we divide our study into two types of factors

(a) $\prod_{i=0}^{\infty}(1+\epsilon_i z)$

(b) $\frac{1}{\prod_{i=0}^{\infty}(1-\eta_i z)}$

In Subsection 2.2.1, we present a procedure to construct superregular matrices from generator polynomials obtained with factors of type (a). In Subsection 2.2.2 we show how the two types of factors are directly related. We prove that a generator polynomial given by the product of factors of type (b) generates a superregular matrix if and only if the corresponding polynomial obtained with factors of type (a) does. In Subsection 2.2.3 we introduce some actions that preserve superregularity.

2.2.1 Factors of type $\prod_{i=0}^{\infty}(1+\epsilon_i z)$

Since superregular matrices are finite, we consider a finite product instead of an infinte one to calculate the generator polynomial

$$g(z)=\prod_{i=0}^{l-1}(1+\epsilon_i z)$$

of the superregular matrix. Let $p(x)$ be an irreducible polynomial of degree n over $\mathbb{F}_{p^n}$ and let α be a root, $p(\alpha)=0$. Then we define $\epsilon_i=\alpha^i$, for $i=0,1,\ldots,l-1$. We compute the generator polynomial

$$g(z)=\prod_{i=0}^{l-1}(1+\alpha^i z)=g_0+g_1 z+\cdots+g_l z^l$$

and we construct the matrix in the following way

$$G=\begin{bmatrix} g_0 & 0 & \cdots & 0 & 0 \\ g_1 & g_0 & \cdots & 0 & 0 \\ \vdots & \vdots & & \vdots & \vdots \\ g_{l-1} & g_{l-2} & \cdots & g_0 & 0 \\ g_l & g_{l-1} & \cdots & g_1 & g_0 \end{bmatrix}. \tag{2.3}$$

Universitat d'Alacant
Universidad de Alicante

This matrix is superregular with the appropriate choice of $p(x)$. Unfortunately, we cannot decide in advance which is the irreducible polynomial that one should take. Due to the fact that the concept of limits over finite fields is not well defined, we cannot translate the condition $\sum_{i=0}^{\infty} \epsilon_i < \infty$ given in expression (2.2) and we cannot decide a priori which are the irreducible polynomials that provide a superregular matrix.

Example 2.1: Let us choose $p(x) = 1 + x^3 + x^4$, an irreducible polynomial over $\mathbb{F} = \mathbb{F}_{2^4}$. If we let α be a root of $p(x)$, we can generate the following polynomial

$$g(z) = (1+z)(1+\alpha z)(1+\alpha^2 z)(1+\alpha^3 z) = 1 + \alpha^6 z + \alpha^2 z^2 + \alpha^9 z^3 + \alpha^6 z^4.$$

The matrix corresponding to this generator polynomial

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ \alpha^6 & 1 & 0 & 0 & 0 \\ \alpha^2 & \alpha^6 & 1 & 0 & 0 \\ \alpha^9 & \alpha^2 & \alpha^6 & 1 & 0 \\ \alpha^6 & \alpha^9 & \alpha^2 & \alpha^6 & 1 \end{bmatrix}$$

is a 5×5 superregular matrix. ■

When working over $\mathbb{F}_p$ the same construction can be used. In this case we choose $\alpha \in \mathbb{F}$, a generator element of the field. In the same way, one cannot predict which generator element should be selected.

Example 2.2: To obtain a 4×4 superregular matrix over $\mathbb{F}_7$ we can use the element $\alpha = 3$ to compute the generator polynomial

$$h(z) = (1+z)(1+3z)(1+3^2 z) = 1 + 6z + 4z^2 + 6z^3.$$

The following 4×4 matrix associated to $h(z)$ is superregular

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 6 & 1 & 0 & 0 \\ 4 & 6 & 1 & 0 \\ 6 & 4 & 6 & 1 \end{bmatrix}.$$

■

Universitat d'Alacant
Universidad de Alicante

Although one cannot predict which are the generator elements or irreducible polynomials that provide a superregular matrix, the computational effort is extremely reduced. In Theorem 2.3 we present an action that preserves the superregularity and again reduces the computer searches by half.

2.2.2 Factors of type $\frac{1}{\prod_{i=0}^{\infty}(1-\eta_i z)}$

The following result proves that if we choose $\eta_i = \beta^i$ for $i = 0, 1, \ldots, l-1$, where β is the root of an irreducible polynomial then the obtained matrix F is superregular if and only if the matrix resultant from the product of factors of type (a), using $\epsilon_i = \beta^i$, is superregular, too.

Lemma 2.1: *Let β be the root of an irreducible polynomial $p(x)$ over $\mathbb{F} = \mathbb{F}_{p^n}$. Then the generator polynomial*

$$f(z) = \frac{1}{\prod_{i=0}^{l-1}(1-\beta^i z)}$$

generates a superregular matrix F if and only if the generator polynomial

$$g(z) = \prod_{i=0}^{l-1}(1+\beta^i z)$$

generates a matrix G that is superregular as well.

Universitat d'Alacant
Universidad de Alicante

PROOF: It is already known that the inverse H^{-1} of an $l \times l$ superregular matrix H is a superregular matrix (see [33]). If $h(z)$ is the polynomial associated to H then we define the l-th truncation of $f(z) = \frac{1}{h(z)}$ in the following way

$$f_{[l]}(z) = f_0 + f_1 z + \cdots + f_l z^l.$$

$f_{[l]}(z)$ is the generator polynomial of H^{-1}. Therefore, to proof our result we will

show that the matrix H generated by the polynomial

$$h(z) = \prod_{i=0}^{l-1}(1 - \beta^i z)$$

is a superregular matrix if and only if the matrix G generated by

$$g(z) = \prod_{i=0}^{l-1}(1 + \beta^i z)$$

is superregular. The conclusion will follow from the fact that $H^{-1} = F$ is a superregular matrix.

Let β be the root of an irreducible polynomial $p(x)$ over $\mathbb{F}_{p^n}$. Then there exist an integer i, with $0 \leq i \leq p^n - 2$, such that $-1 = \beta^i$. We can rewrite the polynomial $h(z)$ in the following way

$$\begin{aligned} h(z) =& (1 - z)(1 - \beta z)(1 - \beta^2 z) \cdots (1 - \beta^{l-1} z) \\ =& (1 + \beta^i z)(1 + \beta^i \beta z)(1 + \beta^i \beta^2 z) \cdots (1 + \beta^i \beta^{l-1} z) \\ =& g_0 + \beta^i g_1 z + (\beta^i)^2 g_2 z^2 + \cdots + (\beta^i)^l g_l z^l. \end{aligned}$$

The minors of the matrix

$$H = \begin{bmatrix} g_0 & 0 & 0 & \cdots & 0 \\ \beta^i g_1 & g_0 & 0 & \cdots & 0 \\ (\beta^i)^2 g_2 & \beta^i g_1 & g_0 & \cdots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ (\beta^i)^l g_l & (\beta^i)^{l-1} g_{l-1} & (\beta^i)^{l-2} g_{l-2} & \cdots & g_0 \end{bmatrix}$$

are the corresponding minors of the matrix G multiplied by factors of β^i. Therefore the not trivially zero minors in H are nonzero if and only if the corresponding minors in G are. In other words, H is a superregular matrix if and only if G is superregular. Since $f(z) = \frac{1}{h(z)}$, its l-th truncation $f_{[l]}(z)$, is the generator polynomial of H^{-1}. Therefore, H^{-1} is a superreguar matrix if and only if H is superregular, thus, if and only if G is superregular. □

Note that over finite fields with characteristic $p = 2$, $h(z) = g(z)$ and the reasoning gets simplified.

Universitat d'Alacant
Universidad de Alicante

We showed how generator polynomials of superregular matrices constructed using factors of type b) are directly related with the class obtained from factors of type a), that is, they have the same behavior. If the root of an irreducible polynomial generates a superregular matrix with factors of class a), it will do with the kind of factors b) too. In section 2.2.3 we show that this case can be seen as a composition of two actions.

2.2.3 Actions preserving superregularity

We already mentioned that operations like the inverse of a matrix preserve superregularity. The following actions preserve this characteristic, too. The first two where introduced in [38], the third is a particular one for the construction presented above.

Theorem 2.1 ([38], Theorem 4.5): *Let A be a superregular matrix over $\mathbb{F} = \mathbb{F}_{p^e}$ and $\alpha \in \mathbb{F}^*$, then*

$$\alpha \bullet A = \begin{bmatrix} a_0 & 0 & 0 & \cdots & 0 \\ \alpha a_1 & a_0 & 0 & \cdots & 0 \\ \alpha^2 a_2 & \alpha a_1 & a_0 & \cdots & \vdots \\ \vdots & \vdots & \vdots & & \vdots \\ \alpha^l a_l & \alpha^{l-1} a_{l-1} & \alpha^{l-2} a_{l-2} & \cdots & a_0 \end{bmatrix}$$

is as well a superregular matrix.

Universitat d'Alacant
Universidad de Alicante

If we translate Theorem 2.1 in terms of our generator polynomial we get that the polynomial associated to $\alpha \bullet A$ is

$$g(z) = \prod_{i=0}^{l-1} (1 + \alpha^{i+1} z) = (1 + \alpha z)(1 + \alpha^2 z) \cdots (1 + \alpha^l z),$$

so in each factor we multiply the term in the variable z by a factor of α. Another way to obtain the superregular matrices generated in Subsection 2.2.2 with factors of type b) is to compose the action in Theorem 2.1 for $\alpha = \beta^i$ and the inverse of a matrix, $(\beta^i \bullet A)^{-1}$.

Theorem 2.2 ([38], Theorem 4.6): *Let A be a superregular matrix over $\mathbb{F} = \mathbb{F}_{p^e}$ and $i \in \mathbb{Z}/e\mathbb{Z}$, then*

$$i \circ A = \begin{bmatrix} a_0^{p^i} & 0 & 0 & \cdots & 0 \\ a_1^{p^i} & a_0^{p^i} & 0 & \cdots & 0 \\ a_2^{p^i} & a_1^{p^i} & a_0^{p^i} & \cdots & \vdots \\ \vdots & \vdots & \vdots & & \vdots \\ a_l^{p^i} & a_{l-1}^{p^i} & a_{l-2}^{p^i} & \cdots & a_0^{p^i} \end{bmatrix}$$

is a superregular matrix.

Theorem 2.3: *Let G and $p(x)$ be as in the construction of Subsection 2.2.1. Then the matrix*

$$K = \begin{bmatrix} k_0 & 0 & 0 & \cdots & 0 & 0 \\ k_1 & k_0 & 0 & \cdots & 0 & 0 \\ k_2 & k_1 & k_0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ k_l & k_{l-1} & k_{l-2} & \cdots & k_1 & k_0 \end{bmatrix}$$

is superregular, where $k(z) = \prod_{i=0}^{l-1}(1 + (\alpha^{-1})^i z) = k_0 + k_1 z + \cdots + k_l z^l$.

Universitat d'Alacant
Universidad de Alicante

PROOF: Let us consider the minors of G and K as polynomials in α and let $\gamma(\alpha)$ denote the minors of G and $\kappa(\alpha)$ be the minors of K. The following relation holds

$$\gamma(\alpha) = \kappa\left(\frac{1}{\alpha}\right).$$

The superregularity of G implies that its minors are nonzero and therefore, the minors of K are different from zero, too. □

The last action again reduces computer searches when using our construction. The roots of an irreducible polynomial $p(x)$ are the inverse of the roots of its reciprocal polynomial $q(x) = x^n p(x^{-1})$, which is to be irreducible as well. So both

polynomials will conduct similarly: if the generator polynomial obtained from a root of $p(x)$ generates a superregular matrix, then so does any root of $q(x)$. This means that only half of the irreducible polynomials in a field need to be tested. For $\mathbb{F} = \mathbb{F}_p$, a generator element and its inverse will perform in the same way.

2.3 Reverse-superregular matrices

In this section we introduce **reverse-superregular matrices**. These are matrices that remain superregular when we invert the position of its elements. This fact provides us with another method to produce a double amount of superregular matrices. Moreover, the 2-direction-superregularity of these matrices has already proved as a very potent tool in the construction of codes which have special forward and backward flexibility, such as reverse-MDP convolutional codes (see Chapter 3). Reverse-superregular matrices point out as very good candidates for obtaining other codes with good characteristics. We prove that the construction proposed in Subsection 2.2.1 generates matrices of this type.

Definition 2.2: A superregular matrix A is a **reverse-superregular** matrix if the matrix

$$A_{\text{rev}} = \begin{bmatrix} a_l & 0 & 0 & \cdots & 0 \\ a_{l-1} & a_l & 0 & \cdots & 0 \\ a_{l-2} & a_{l-1} & a_l & \cdots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ a_0 & a_1 & a_2 & \cdots & a_l \end{bmatrix}$$

is superregular.

Universitat d'Alacant
Universidad de Alicante

The generator polynomial of A_{rev} is $\overline{g}(z) = z^l g(\frac{1}{z}) = a_l + a_{l-1}z + \cdots + a_0 z^l$.

The following example shows that these matrices can be harder to find since this is a more restrictive condition than simple superregularity.

Example 2.3: Not every superregular matrix is a reverse-superregular matrix. For example, the following 5×5 matrix

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 2 & 1 & 1 & 0 & 0 \\ 6 & 2 & 1 & 1 & 0 \\ 2 & 6 & 2 & 1 & 1 \end{bmatrix}$$

is superregular over $\mathbb{F}_7$. However, its reverse matrix

$$A_{\text{rev}} = \begin{bmatrix} 2 & 0 & 0 & 0 & 0 \\ 6 & 2 & 0 & 0 & 0 \\ 2 & 6 & 2 & 0 & 0 \\ 1 & 2 & 6 & 2 & 0 \\ 1 & 1 & 2 & 6 & 2 \end{bmatrix}$$

is not superregular since the minor formed extracting the rows 2, 3, 5 and the columns 1, 2, 3 is zero

$$\begin{vmatrix} 6 & 2 & 0 \\ 2 & 6 & 2 \\ 1 & 1 & 2 \end{vmatrix} = 0.$$

■

Universitat d'Alacant
Universidad de Alicante

Although these matrices might be hard to find, Example 2.4 reveals that we are not as restricted as one could think.

Example 2.4: It is clear that superregular matrices that are symmetric with respect to a lower diagonal are reverse-superregular, like in this matrix

$$B = \begin{bmatrix} 1 & 0 & 0 & 0 \\ \alpha^2 & 1 & 0 & 0 \\ \alpha^2 & \alpha^2 & 1 & 0 \\ 1 & \alpha^2 & \alpha^2 & 1 \end{bmatrix}.$$

B is superregular over $\mathbb{F} = \mathbb{F}_{2^3}$, where $\alpha^3 + \alpha + 1 = 0$. Since $B = B_{\text{rev}}$, B is reverse-superregular. But the following matrix shows us that this is not a necessary

condition.

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ \alpha & 1 & 0 & 0 \\ \alpha^3 & \alpha & 1 & 0 \\ \alpha^5 & \alpha^3 & \alpha & 1 \end{bmatrix} = C \neq C_{\text{rev}} = \begin{bmatrix} \alpha^5 & 0 & 0 & 0 \\ \alpha^3 & \alpha^5 & 0 & 0 \\ \alpha & \alpha^3 & \alpha^5 & 0 \\ 1 & \alpha & \alpha^3 & \alpha^5 \end{bmatrix}$$

and both are superregular matrices. So C is a reverse-superregular matrix over $\mathbb{F}_{2^3}$. ■

The following theorem proves that the construction we propose in Subsection 2.2.1 gives a very efficient mechanism to generate reverse-superregular matrices.

Theorem 2.4: *Let $p(x)$ be an irreducible polynomial of degree n over $\mathbb{F}_{p^n}$ and let α be a root, $p(\alpha) = 0$. Let $r(z) = \prod_{i=0}^{l-1}(1 + \alpha^i z) = r_0 + r_1 z + \cdots + r_l z^l$ be a generator polynomial. If the matrix associated to $r(z)$, that is,*

$$R = \begin{bmatrix} r_0 & 0 & \cdots & 0 \\ r_1 & r_0 & \ddots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ r_l & \cdots & r_1 & r_0 \end{bmatrix}$$

is superregular, then the reverse matrix R_{rev} is superregular (R is a reverse-superregular matrix).

Universitat d'Alacant
Universidad de Alicante

PROOF: By construction we know that

$$r_0 = 1, \quad r_1 = \sum_{j=0}^{l-1} \alpha^j, \quad r_2 = \sum_{0 \leq j < k \leq l-1} \alpha^j \alpha^k,$$

$$r_3 = \sum_{0 \leq j < k < h \leq l-1} \alpha^j \alpha^k \alpha^h, \quad \ldots, \quad r_n = \prod_{j=0}^{l-1} \alpha^j.$$

Let $\rho(\alpha)$ be the minors of R and denote by $\overline{\rho}(\alpha)$ the minors of R_{rev}. They are related in the following way

$$\overline{\rho}(\alpha) = \alpha^t \rho\left(\frac{1}{\alpha}\right),$$

where the power t depends on the size of the minor. Then it is clear that if the minors of R are different from zero, so are the minors of R_{rev}. □

In this case computer searches for reverse-superregular matrices are reduced since superregularity only needs to be tested once.

The actions presented in Theorems 2.1, 2.2 and 2.3 preserve reverse-superregularity as well. In Theorems 2.1 and 2.2 the minors of A_{rev} are only transformed by factors of α^i or powers of p^i, so the property is maintained. If we apply Theorem 2.3 to matrix the G_{rev}, then the minors of G_{rev}, $\overline{\gamma}(\alpha)$, and the ones of K_{rev}, $\overline{\kappa}(\alpha)$, still hold the relation

$$\overline{\gamma}(\alpha) = \overline{\kappa}\left(\frac{1}{\alpha}\right)$$

so Theorem 2.3 preserves reverse-superregularity, too.

However, not all actions preserving superregularity preserve reverse-superregularity, as we show in the last example.

Example 2.5: The inverse of a superregular matrix is a superregular matrix, too, but the same does not occur with reverse-superregularity. This happens because the reversed matrix of the inverse is not necessarily a superregular matrix.

The following matrix is reverse-superregular over $\mathbb{F} = \mathbb{F}_{2^4}$ with $1 + \alpha^3 + \alpha^4 = 0$

$$Y = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ \alpha^6 & 1 & 0 & 0 & 0 \\ \alpha^2 & \alpha^6 & 1 & 0 & 0 \\ \alpha^9 & \alpha^2 & \alpha^6 & 1 & 0 \\ \alpha^6 & \alpha^9 & \alpha^2 & \alpha^6 & 1 \end{bmatrix}.$$

However its inverse is not a reverse-superregular matrix since in

$$(Y^{-1})_{\text{rev}} = \begin{bmatrix} \alpha^6 & 0 & 0 & 0 & 0 \\ \alpha^{11} & \alpha^6 & 0 & 0 & 0 \\ \alpha^7 & \alpha^{11} & \alpha^6 & 0 & 0 \\ \alpha^6 & \alpha^7 & \alpha^{11} & \alpha^6 & 0 \\ 1 & \alpha^6 & \alpha^7 & \alpha^{11} & \alpha^6 \end{bmatrix}$$

the minor

$$\begin{vmatrix} \alpha^{11} & \alpha^{6} & 0 \\ \alpha^{7} & \alpha^{11} & \alpha^{6} \\ \alpha^{6} & \alpha^{7} & \alpha^{11} \end{vmatrix} = 0.$$

■

2.4 Necessary and sufficient field sizes: a comparison

In this section we give an overview of the minimum field sizes necessary for superregularity and reverse-superregularity and compare them with the ones obtained in our construction, which are just sufficient to find a superregular or reverse-superregular matrix. The tables mentioned in this section can be found at the end of the chapter.

In Table 2.3 we present the minimum field sizes necessary to find a superregular and a reverse-superregular matrix obtained by computer search. The fields for reverse-superregularity are larger than the ones obtained for superregularity since, as we explained before, reverse-superregularity is a more restrictive condition. One can notice that the minimum field size for reverse-superregularity grows faster.

If we compare the field sizes obtained by the construction given in [33] for superregular matrices versus the ones obtained for our construction we see that we obtain smaller field sizes, closer to the minimum necessary field size computed. This can be observed in Table 2.4.

Using the construction proposed here to generate matrices over $\mathbb{F} = \mathbb{F}_{p^n}$ one can check the triangular scheme that we obtain in Table 2.5. Although it is not proved, the trend indicates that one could always generate this kind of $l \times l$ matrices over $\mathbb{F}_{p^n}$ using a field size of $p^n = q^2$, for q the $(l-3)$-th prime. If this conjecture is right this would beat the one done in [33]. In [33] it was conjectured that there always exists an $l \times l$ superregular matrix over $\mathbb{F}_{2^{l-2}}$. But for $l \geq 11$ we know that $q^2 < 2^{l-2}$, as one can see in Table 2.1. In Table 2.6 the irreducible polynomials that generate reverse-superregular matrices using our construction over fields of characteristic $p = 2$ are presented. In the last column one can find the reciprocal of

those polynomials, which generate reverse-superregular matrices, too. In Table 2.7 a similar description is made for fields with characteristic $p = 3$.

In Table 2.2 we detail the generator elements that one should choose over $\mathbb{F}_p$ in order to obtain reverse-superregular matrices using our construction. The corresponding inverse elements that also generate reverse-superregular matrices can be found in the last column. The field sizes shown in the table are the first ones where a reverse-superregular matrix of that size can be generated using the procedure proposed in this work. For instance, over $\mathbb{F}_{13}$ a 5×5 reverse-superregular matrix can be constructed but not one of size 6×6.

It is clear that if an $l \times l$ reverse-superregular matrix can be obtained over $\mathbb{F}_p$ or $\mathbb{F}_{p^n}$ using our method, then $j \times j$ matrices for $j \leq l$ can be generated too.

One should notice that the field size is always smaller if we generate matrices over $\mathbb{F}_p$ than if we do it over $\mathbb{F}_{p^n}$.

In the Table 2.8, and as an interesting case to study, we present the results of a modification on our method. Instead of multiplying $l - 1$ factors in order to obtain the l elements of an $l \times l$ matrix, one can multiply many more factors and truncate the obtained polynomial at its l-th iteration. In this way the appearance of superregular matrices in smaller fields is provoked. These new field sizes are much closer to the necessary minimum than the ones observed in Table 2.4. In the 3rd column we indicate how many factors we need to multiply in order that the obtained matrix is superregular. As an effect of these added multiplications the reverse-superregularity of the matrix is not guaranteed anymore. It is not clear how one can predict this behavior and decide in advance how many factors should be multiplied to obtain a superregular matrix. The study of this occurences could help to reduce the minimum field sizes in the previous constructions.

2.5 Conclusions

In this paper, we propose a method for the generation of superregular matrices. We use similar techniques as the ones for obtaining totally positive sequences. We demonstrate that the different types of factors used for the generation of these sequences are indeed reduced to only one type. We give some actions preserving superregularity in general and a specific one for our construction. Moreover, we prove

that the matrices generated in this way are actually more sophisticated, having twice the superregular property. We call these matrices reverse-superregular matrices since they satisfy the property in the inverted direction.

In the last section we compare the field sizes given by the different constructions with the minimum ones calculated using computer search programs. We show how our construction can generate superregular matrices using smaller fields than the ones obtained in previous proposed constructions and we point out a conjecture that could overcome the threshold proposed in a previous conjecture done in [33].

l	5	6	7	8	9	10	11	12	13
q^2	9	25	49	121	169	289	361	529	841
2^{l-2}	8	16	32	64	128	256	512	1024	2048

Table 2.1: *Comparison between q^2 and 2^{l-2}*

Field size	Matrix size	Generator element	Inverse element
5	3×3	2	3
7	4×4	3	5
11	5×5	2	6
		7	8
17	6×6	5	7
37	7×7	2	19
73	8×8	13	45
149	9×9	3	50

Table 2.2: *Generator elements that generate reverse-superregular matrices over $\mathbb{F}_p$ using our method*

Universitat d'Alacant
Universidad de Alicante

l	Min. field size superregularity	Min. field size rev-superregularity
4×4	5	5
5×5	7	7
6×6	11	13
7×7	17	19
8×8	31	59
9×9	59	≤ 149
10×10	≤ 127	

Table 2.3: *Minimum necessary field sizes*

Matrix size	Construction in [33]	Our construction
4×4	7	7
5×5	11	11
6×6	23	17
7×7	43	37
8×8	79	73
9×9	211	149

Universitat d'Alacant
Universidad de Alicante

Table 2.4: *Minimum field sizes obtained with the construction given in [33] and with our method over* $\mathbb{F}_p$

l	$\mathbb{F}_{2^n}$	$\mathbb{F}_{3^n}$	$\mathbb{F}_{5^n}$	$\mathbb{F}_{7^n}$	$\mathbb{F}_{11^n}$	$\mathbb{F}_{13^n}$
4×4	$2^3 = 8$					
5×5	$2^4 = 16$	$3^2 = 9$				
6×6	$2^5 = 32$	$3^3 = 27$	$5^2 = 25$			
7×7	$2^6 = 64$	$3^4 = 81$	$5^3 = 125$	$7^2 = 49$		
8×8	$2^7 = 128$	$3^5 = 243$	$5^4 = 625$	$7^3 = 343$	$11^2 = 121$	
9×9	$2^8 = 256$	$3^6 = 729$	$5^5 = 3125$	$7^4 = 2401$	$11^3 = 1331$	$13^2 = 169$

Table 2.5: *Field sizes obtained with our construction over* $\mathbb{F}_{p^n}$

Field size	Matrix size	Irreducible polynomial	Reciprocal polynomial
2^3	4×4	x^3+x+1	x^3+x^2+1
2^4	5×5	x^4+x+1	x^4+x^3+1
2^5	6×6	x^5+x^2+1	x^5+x^3+1
		$x^5+x^3+x^2+x+1$	$x^5+x^4+x^3+x^2+1$
2^6	7×7	$x^6+x^5+x^2+x+1$	$x^6+x^5+x^4+x+1$
		$x^6+x^5+x^4+x^2+1$	$x^6+x^4+x^2+x+1$
2^7	8×8	$x^7+x^6+x^3+x+1$	$x^7+x^6+x^4+x+1$
		x^7+x^6+1	x^7+x+1
		$x^7+x^6+x^5+x^4$ $+x^2+x+1$	$x^7+x^6+x^5+x^3$ $+x^2+x+1$
		$x^7+x^6+x^4+x^2+1$	$x^7+x^5+x^3+x+1$
2^8	9×9	$x^8+x^6+x^5+x^4$ $+x^3+x+1$	$x^8+x^7+x^5+x^4$ $+x^3+x^2+1$
		$x^8+x^6+x^5+x^4$ $+x^2+x+1$	$x^8+x^7+x^6+x^4$ $+x^3+x^2+1$
		$x^8+x^7+x^6+x^5$ $+x^4+x+1$	$x^8+x^7+x^4+x^3$ $+x^2+x+1$
		$x^8+x^5+x^4+x^3$ $+x^2+x+1$	$x^8+x^7+x^6+x^5$ $+x^4+x^3+1$
		$x^8+x^6+x^4+x^3$ $+x^2+x+1$	$x^8+x^7+x^6+x^5$ $+x^4+x^2+1$
		$x^8+x^6+x^5+x+1$	$x^8+x^7+x^3+x^2+1$
		$x^8+x^7++x^5+x^3+1$	$x^8+x^5+x^3+x+1$
		$x^8+x^4+x^3+x^2+1$	$x^8+x^6+x^5+x^4+1$
		$x^8+x^4+x^3+x+1$	$x^8+x^7+x^5+x^4+1$

Table 2.6: *Irreducible polynomials that generate reverse-superregular matrices over $\mathbb{F}_{2^n}$ using our method*

Universitat d'Alacant
Universidad de Alicante

Field size	Matrix size	Irreducible polynomial	Reciprocal polynomial
3^2	5×5	$x^2 + x + 2$	$x^2 + 2x + 2$
3^3	6×6	$x^3 + 2x^2 + 1$	$x^3 + 2x + 1$
		$x^3 + x^2 + x + 2$	$x^3 + 2x^2 + 2x + 2$
		$x^3 + x^2 + 2x + 1$	$x^3 + 2x^2 + x + 1$
3^4	7×7	$x^4 + 2x^3 + 2$	$x^4 + x + 2$
		$x^4 + x^3 + 2x^2 + 2x + 2$	$x^4 + x^3 + x^2 + 2x + 2$
		$x^4 + x^3 + 2$	$x^4 + 2x + 2$
		$x^4 + x^2 + 2$	$x^4 + 2x^2 + 2$
		$x^4 + x^3 + 2x + 1$	$x^4 + 2x^3 + x + 1$
3^5	8×8	$x^5 + x^3 + x^2 + 2$	$x^5 + 2x^3 + 2x^2 + 2$
		$x^5 + x^4 + x^3$ $+x^2 + 2x + 1$	$x^5 + 2x^4 + x^3$ $+x^2 + x + 1$
		$x^5 + 2x + 2$	$x^5 + x^4 + 2$
		$x^5 + x^4 + x^2 + x + 1$	$x^5 + x^4 + x^3 + x + 1$
		$x^5 + 2x^4 + x^3$ $+x^2 + x + 2$	$x^5 + 2x^4 + 2x^3$ $+2x^2 + x + 2$
		$x^5 + 2x^4 + x^3$ $+2x^2 + 2x + 2$	$x^5 + x^4 + x^3$ $+2x^2 + x + 2$
		$x^5 + 2x^4 + 2x^3 + x^2 + 1$	$x^5 + x^3 + 2x^2 + 2x + 1$
		$x^5 + 2x^4 + x + 1$	$x^5 + x^4 + 2x + 1$
		$x^5 + x^4 + x^3$ $+2x^2 + x + 1$	$x^5 + x^4 + 2x^3$ $+x^2 + x + 1$
		$x^5 + 2x^4 + 2x^3 + x^2 + 2$	$x^5 + 2x^3 + x^2 + x + 2$

Table 2.7: *Irreducible polynomials that generate reverse-superregular matrices over $\mathbb{F}_{3^n}$ using our method*

Universitat d'Alacant
Universidad de Alicante

Field size	Matrix size	Number of factors	Generator element	Inverse element
5	3×3	2	2	3
7	4×4	3	3	5
11	5×5	4	2	6
			7	8
17	6×6	5	5	7
		8	3	5
		$6, 10$	10	12
		8	11	14
23	7×7	11	15	20
		11	17	19
41	8×8	20	24	12
107	9×9	$9, 97$	26	70

Table 2.8: *First appearance of superregular matrices with different number of multiplications*

Universitat d'Alacant
Universidad de Alicante

Universitat d'Alacant
Universidad de Alicante

Chapter 3

Performance of convolutional codes over the erasure channel

3.1 Introduction

The erasure channel was introduced by Elias [20] in 1955 as a toy example of communication channel. It was not until 40 years later, with the development of the Internet, when it was considered more than a theoretical channel model.

The erasure channel is a communication channel where each sent symbol is either correctly received or considered as erased; this can happen because the symbol was corrupted or simply not received. In the concrete case of the binary erasure channel (BEC) each bit is erased with probability π, while it is received correctly with probability $1-\pi$, where $0 < \pi < 1$. If we choose a larger alphabet, $\mathbb{F}_q$ with $q > 2$, then the situation is as illustrated in Figure 3.1, where X represents an erased symbol.

Depending on the communication scenario these erasures occur due to different reasons, but it is quite usual that they are clustered together in groups or bursts. This is a phenomena observed about many channels modeled via the erasure channel. This means that the erasure probability of a symbol is not constant and it increases after one erasure has already occurred, in other words, the chance that another erasure occurs right after one symbol is erased increases. Therefore, a much better way to model this situation is by means of a first order Markov chain as shown in Figure 3.2, where $0 < \pi < \theta < 1$ and b represents a symbol of the alphabet.

This behavior can be observed, for instance, in optical or magnetic storage de-

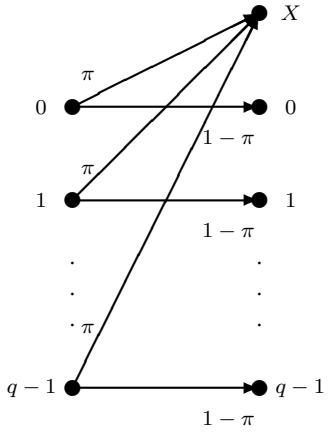


Figure 3.1: *Erasure channel over large alphabet.*

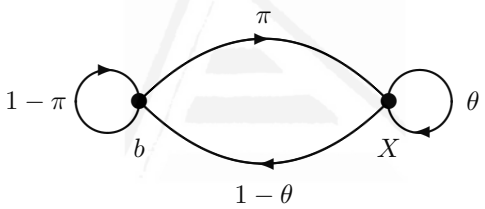


Figure 3.2: *Representation of the erasure channel as a Markov chain.*

Universitat d'Alacant
Universidad de Alicante

vices like CDs, DVDs or hard disks, where scratches or thermal acerbities can make unreadable contiguous blocks of symbols generating the above mentioned bursts (see, e.g., [78, 88]). Another example are space communication systems where incorrectly decoded frames of data are removed from the information stream before being handed up to higher layers, and the result is a sequence of correctly decoded data containing bursts of erasures (see, e.g., [4, 10]).

The channel that can better illustrate this situation is the Internet. On the Internet, the information is transmitted in the form of packets. Since these packets have headers and number sequences describing their position within a given stream, the receiver knows where the erasures have happened. Packet sizes are upper bounded by 12,000 bits - the maximum that the Ethernet protocol allows (that everyone uses at the user end). In many cases, this maximum is actually used. Due to the nature of the TCP part of the TCP/IP protocol stack, most sources need an acknowledg-

ment confirming that the packet has arrived at the destination; these packets are only 320 bits long. So if everyone were to use TCP/IP, the packet size distribution would be as follows: 35% – 320 bits, 35% – 12,000 bits and 30% – in between the two, uniform. Real-time traffic used, e.g., in video calling, does not need an acknowledgment since that would take too much time; overall, the following is a good assumption of the packet size distribution: 30% – 320 bits, 50% – 12,000 bits, 20% – in between, uniform. This explains that the alphabet size we use to model the Internet is normally very big. We can represent each packet as an element or sequence of elements from the alphabet, for instance, the finite field $\mathbb{F} = \mathbb{F}_{2^{1,000}}$. If a packet has less than 1,000 bits then one uses simply the corresponding element of $\mathbb{F}$. If the packet is larger one uses several alphabet symbols to describe the packet.

The occurrence of bursts over the Internet is mainly provoked by buffer overflows at routers due to a continuous data flow, such as HTTP or FTP, and bit errors caused by noise, usually tested using cyclic redundancy check (CRC) codes.

The growing number of users of this channel has made the Internet one of the most important communication infrastructure nowadays. However, this increase of the traffic volume makes difficult to maintain the quality of communication. When congestion occurs, users see it as a delay in the response time of applications and this is due to the TCP/IP protocol used on the Internet. In order to warrant reliable transmission of data the TCP/IP protocol applies the retransmission of lost packets [64]. This ensures the reception of the information by the receiver, however it enlarges the transmission time and has unsatisfactory performance when data need to be transmitted from one server to multiple receivers or in real-time applications like video-calling.

As a consequence, other type of methods have been proposed in order to avoid these retards [47, 76, 91]. Forward error correction is a coding based solution. Before transmission, data are encoded using some linear error correcting code which will later allow us to correct as many erasures as possible based on its special structural properties.

Until now mainly block codes have been used for such a task. Luby [54] proposed a first realization of a class of erasure codes called LT codes. These are rateless codes generated on the fly as needed. Shokrollahi, Lasse and Luby [77] added an outer code to LT codes and introduced the so-called Raptor codes, studying its behav-

ior over the erasure channel. Low-density-parity-check (LDPC) codes have been studied for application on the BEC as well [55, 66]; degree distributions achieving the channel capacity have been introduced (see, e.g., [63, 76]). In [1], a generalized iterative decoder valid for any linear block code and adaptable to system specifications was presented. Moreover, the consultative Committee for Space Data Systems (CCSDS), motivated by the need of efficient solutions to the burst erasure channel problem, studied the design of irregular repeat-accumulate (IRA) codes, generalized IRA codes, Tornado codes and protograph-based codes (see, e.g., [10, 16, 55]). However, the use of convolutional codes over this type of channel has been much less studied.

In this chapter we propose convolutional codes as an alternative to block codes over the erasure channel and we provide a more detailed development than the ones made untill now (see, e.g., [6, 21, 50]). In this chapter we use the module theoretic representation introduced in Section 1.3 and we understand the received symbols as a polynomial, so we look at it as if it was a complete sequence and we can slide up and down along it. This is known as the *sliding window* characteristic and provides convolutional codes with much more flexibility than block codes. The received information can be grouped in blocks or windows in many ways, depending on the erasure bursts, and then be decoded by decoding the *easy* blocks first. This flexibility in grouping information brings certain freedom in the handling of sequences; we can split the blocks in smaller windows, we can overlap windows, etc., we can proceed to decode in a less strict order. The blocks are not fixed as in the block code case, i.e., they do not have a fixed grouping of a fixed length. We can slide along the transmitted sequence and decide the place where we want to start our decoding. In oder words, we can adapt the process to the pattern of erasures we receive. With this property we are able to correct in a given block more erasures than a block code of that same length could do.

An $[N, K]$ block code used for transmission over an erasure channel can correct up to $N - K$ erasures in a block of length N. The optimal recovering capability of $N - K$ is achieved by an MDS $[N, K]$-code. In this chapter we prove that an (n, k, δ)-convolutional code can recover $d_j^c(\mathcal{C}) - 1$ erasures in windows of size $n(j+1)$. The subclass of convolutional codes that can recover a maximal amount of erasures per window size is that of MDP convolutional codes. We analyze the behavior of MDP codes and we move one step forward to improve the recovering process. To do

so we construct a new type of codes called reverse-MDP convolutional codes. These codes are able to invert the direction of the decoding to be done from right-to-left. Due to this fact one can recover through a backward process erasures that would be already considered as lost by an MDP code. We prove the existence of reverse-MDP codes and give a particular construction based on the reverse-superregular matrices introduced in Chapter 2. As a final step we add stronger and more restrictive conditions to reverse-MDP codes in order to achieve an optimum performance. We obtain what we call complete-MDP convolutional codes. These codes help to reduce the waiting time produced when a large burst of erasures occurs and we get lost in the recovering process. Simulations results show that these codes can decode extremely efficiently when compared to MDS block codes. Thus, they provide a very attractive choice when transmitting over the erasure channel.

3.2 Decoding over an erasure channel

Let us suppose that we use a convolutional code $\mathcal{C}$ to transmit over an erasure channel. Then we can state the following result.

Theorem 3.1: *Let $\mathcal{C}$ be an (n,k,δ)-convolutional code with $d_{j_0}^c$ the j_0-th column distance. If in any sliding window of length $(j_0+1)n$ at most $d_{j_0}^c-1$ erasures occur then we can recover completely the transmited sequence.*

Universitat d'Alacant
Universidad de Alicante

PROOF: Assume that we have been able to correctly decode up to an instant $t-1$. Then we have the following homogeneous system:

$$\begin{bmatrix} H_\nu & H_{\nu-1} & \cdots & H_{\nu-j_0} & \cdots & H_0 & & & \\ & H_\nu & \cdots & H_{\nu-j_0+1} & \cdots & H_1 & H_0 & & \\ & & \ddots & & & & & \ddots & \\ & & & H_\nu & \cdots & H_{j_0} & H_{j_0-1} & \cdots & H_0 \end{bmatrix} \begin{bmatrix} \boldsymbol{v}_{t-\nu} \\ \vdots \\ \boldsymbol{v}_{t-1} \\ \star \\ \star \\ \vdots \\ \star \end{bmatrix} = \mathbf{0} \qquad (3.1)$$

where $\star$ takes the place of a vector that had some of the components erased. Let

the positions of the erased field elements be $i_1, \ldots, i_e$, where $i_1, \ldots, i_s$, $s \leq n$, with $s \leq n \leq e \leq d^c_{j_0}$, are the erasures occurring in the first n-vector erased. We can take the columns of the matrix in equation (3.1) that correspond to the coefficients of the erased elements to be the coefficients of a new system. The rest of the columns in (3.1) will help us to compute the independent terms. In this way we get a nonhomogeneous system with $(j_0+1)(n-k)$ equations and e, at most $d^c_{j_0}-1$, variables.

We claim that there is an extension $\{\tilde{\boldsymbol{v}}_t, \ldots, \tilde{\boldsymbol{v}}_{t+j_0}\}$ such that the vector

$$(\boldsymbol{v}_{t-\nu}, \ldots, \boldsymbol{v}_{t-1}, \tilde{\boldsymbol{v}}_t, \ldots, \tilde{\boldsymbol{v}}_{t+j_0})$$

is a codeword and such that $\tilde{\boldsymbol{v}}_t$ is unique.

Indeed, we know that a solution of the system exists since we assumed that only erasures occur. To prove the uniqueness of $\tilde{\boldsymbol{v}}_t$, or equivalently, of the erased elements $\tilde{v}_{i_1}, \ldots, \tilde{v}_{i_s}$, let us suppose there exist two such good extensions $\{\tilde{\boldsymbol{v}}_t, \ldots, \tilde{\boldsymbol{v}}_{t+j_0}\}$ and $\{\tilde{\tilde{\boldsymbol{v}}}_t, \ldots, \tilde{\tilde{\boldsymbol{v}}}_{t+j_0}\}$. Let $\boldsymbol{h}_{i_1}, \ldots, \boldsymbol{h}_{i_e}$, be the column vectors of the sliding parity-check matrix in (3.1) which correspond to the erasure elements. We have

$$\tilde{v}_{i_1}\boldsymbol{h}_{i_1} + \cdots + \tilde{v}_{i_s}\boldsymbol{h}_{i_s} + \cdots + \tilde{v}_{i_e}\boldsymbol{h}_{i_e} = \tilde{\boldsymbol{b}}$$

and

$$\tilde{\tilde{v}}_{i_1}\boldsymbol{h}_{i_1} + \cdots + \tilde{\tilde{v}}_{i_s}\boldsymbol{h}_{i_s} + \cdots + \tilde{\tilde{v}}_{i_e}\boldsymbol{h}_{i_e} = \tilde{\tilde{\boldsymbol{b}}},$$

where the vectors $\tilde{\boldsymbol{b}}$ and $\tilde{\tilde{\boldsymbol{b}}}$ correspond to the known part of the system. Subtracting these equations and observing that $\tilde{\boldsymbol{b}} = \tilde{\tilde{\boldsymbol{b}}}$, we obtain:

$$(\tilde{v}_{i_1} - \tilde{\tilde{v}}_{i_1})\boldsymbol{h}_{i_1} + \cdots + (\tilde{v}_{i_s} - \tilde{\tilde{v}}_{i_s})\boldsymbol{h}_{i_s} + \cdots + (\tilde{v}_{i_e} - \tilde{\tilde{v}}_{i_e})\boldsymbol{h}_{i_e} = \boldsymbol{0}.$$

Using Theorem 1.2 we obtain that, necessarily,

$$\tilde{v}_{i_1} - \tilde{\tilde{v}}_{i_1} = 0, \quad \ldots, \quad \tilde{v}_{i_s} - \tilde{\tilde{v}}_{i_s} = 0,$$

This concludes the proof of our claim.

In order to find the value of this unique vector, we solve the full column rank system, find a solution and retain the part which is unique. Then we slide n bits to the next $n(j_0+1)$ window and proceed as above. □

Universitat d'Alacant
Universidad de Alicante

The best scenario happens when the convolutional code is MDP. In this case full error correction 'from left to right' is possible as soon as the fraction of erasures is not more than $\frac{n-k}{n}$ in any sliding window of length $(L+1)n$.

Corollary 3.1: *Let $\mathcal{C}$ be an MDP (n,k,δ)-convolutional code. If in any sliding window of length $(L+1)n$ at most $(L+1)(n-k)$ erasures occur in a transmitted sequence then we can completely recover the sequence in polynomial time in δ.*

The decoding algorithm requires only simple linear algebra. In the optimum case of an MDP convolutional code, for every $(L+1)(n-k)$ erasures a matrix of size at most $(L+1)(n-k)$ has to be inverted over the base field $\mathbb{F}$. This is easily achieved even over fairly large fields.

Theorem 3.1 is optimal in a certain sense: One can show that for any (n,k,δ)-convolutional code there exist patterns of $(L+2)(n-k)$ erasures in a sliding window of length $(L+2)n$ which cannot be uniquely decoded.

One must notice that although in Theorem 3.1 we fix the value $j=j_0$, other sizes of window can be taken during the process in order to optimize it. For any value of j, at most d_j^c-1 erasures can be recovered in a window of size $(j+1)n$. In the MDP case, the parameter L gives an upper bound on the length of the window we can take to correct. For every $j \leq L$ in a window of size $(j+1)n$ we can recover at most $(j+1)(n-k)$ erasures. This means that we can conveniently choose the size of the window we need at each step depending on the distribution of the erasures in the sequence. This is an advantage of these codes over block codes. If we receive a part of sequence with a few errors we do not need to wait until we receive the complete block, we can already decode after very small windows.

This property allows us to recover the erasures in situations where the MDS block codes cannot do it. The following example illustrates this scenario. We compare an MDP convolutional code with an MDS block code of the same length as the maximum window size taken for the convolutional code.

Example 3.1: Let us take an MDP $(2,1,50)$-convolutional code to decode over an erasure channel. In this case the decoding can be completed if in any sliding window of length 202 there are not more than 101 erasures; 50% of the erasures can be recovered.

The MDS block code which achieves a comparable performance is a MDS $[202, 101]$ block code. In a block of 202 symbols we can recover 101 erasures, that is again 50%.

Assume that we use the previously mentioned codes to transmit over an erasure channel. Suppose we have been able to correctly decode up to an instant t and then we receive the following pattern of erasures

$$\ldots vv|\overbrace{\star\star\ldots\star\star}^{(A)60}\overbrace{vv\ldots v}^{(B)80}\overbrace{\star\star\ldots\star\star}^{(C)60}vv|vv\ldots,$$

where each $\star$ stands for a component of the vector that has been erased and v means that the component has been correctly received. In this situation 120 erasures happen in a block of 202 symbols and the MDS block code is not able to recover them. In the block code situation one has to skip the whole window losing that information, and go on with the decoding of the next block.

However, the MDP convolutional code can deal with this situation. Let us frame a 120 symbols length window; in this window we can correct up to 60 erasures. We can take 100 previously decoded symbols, then frame a window with the first 60 erasures from A and 60 more clean symbols from B. In this way we can recover the first block of erasures.

$$\overbrace{vv\ldots vv}^{100}|\overbrace{\star\star\ldots\star\star}^{(A)60}\overbrace{vv\ldots v}^{(B)60}$$

Then we slide through the received sequence of 120 symbols window until we frame the rest of the erasures in the same way.

$$\overbrace{vv\ldots vv}^{(A+B)100}\overbrace{\star\star\ldots\star\star}^{(C)60}\overbrace{vv|vv\ldots}^{60}$$

After this we have correctly decoded the sequence. ■

Remark 3.1: There are situations in which the pattern of erasures that is mentioned in Theorem 3.1 or in Corollary 3.1 does not happen nor can not recover choosing smaller window sizes. Then we get lost in the recovering process.

When using the parity check matrix we know that

$$\begin{bmatrix} H_\nu & H_{\nu-1} & \cdots & H_{\nu-j} & \cdots & H_0 & & & \\ & H_\nu & \cdots & H_{\nu-j+1} & \cdots & H_1 & H_0 & & \\ & & \ddots & & & & & \ddots & \\ & & & H_\nu & \cdots & H_j & H_{j-1} & \cdots & H_0 \end{bmatrix} \begin{bmatrix} \boldsymbol{v}_{t-\nu} \\ \vdots \\ \boldsymbol{v}_t \\ \boldsymbol{v}_{t+1} \\ \vdots \\ \boldsymbol{v}_{t+j} \end{bmatrix} = \mathbf{0}.$$

Then, in order to continue our recovering process once we get lost, we need to find a block of νn symbols without erasures ($\boldsymbol{v}_{t-\nu}$ to $\boldsymbol{v}_{t-1}$) preceding a block of $(j+1)n$ symbols ($\boldsymbol{v}_t$ to $\boldsymbol{v}_{t+j}$) where not more that $d_j^c - 1$ erasures occur, or $d_j^c - 1 = (j+1)(n-k)$ in the case of MDP codes. In other words, we need to have *clean memory*. In that way we can solve again our system and recover $\boldsymbol{v}_t$ to $\boldsymbol{v}_{t+j}$.

We define the *recovering rate per window* as $R_\omega = \frac{\#\text{erasures recovered}}{\#\text{symbols in a window}}$. One should notice that the above mentioned condition to go back to the recovering process is a sufficient condition for R_ω to be maintained. For any generic (n, k, δ)-convolutional code $R_\omega = \frac{d_j^c - 1}{(j+1)n}$. In the MDP case when the number of possible recovered erasures is maximized, we have $R_\omega = \frac{(j+1)(n-k)}{(j+1)n}$. ■

Universitat d'Alacant
Universidad de Alicante

3.3 The backward process and reverse-MDP convolutional codes

In this section we introduce a new class of codes called **reverse-MDP convolutional codes** which have the MDP property both forward and backward, i.e., if we truncate sequences with $\boldsymbol{v}_0 \neq \mathbf{0}$ either in the beginning to obtain $[\boldsymbol{v}_0, \ldots, \boldsymbol{v}_L]$ or at the end of a length L window, for $\boldsymbol{v}_L \neq \mathbf{0}$, the minimum weight of the segment obtained is as large as possible. This fact will help us to solve situations for which an MDP code fails.

In Remark 3.1 we showed necessary conditions to restart recovering once we get lost, i.e., once the number of erasures is too large finding a window of clean or "almost" clean memory. However, finding a whole block of clean memory can end up in a long waiting time, together with a loss of a long part of the sequence until a

νn block of clean symbols is found. The following example illustrates this situation.

Example 3.2: As previously, assume we use a $(2, 1, 50)$ MDP convolutional code to transmit over an erasure channel. Suppose that we are able to recover the sequence up to an instant t, after which we receive a part of a sequence with the following pattern

$$\ldots \star\star\overbrace{\star\ldots\star}^{(A)22}\ \overbrace{vv\star\star vv\star\star\ldots vv\star\star}^{(B)180}\ |\overbrace{vv\ldots vv}^{(C)202}|$$

$$|\overbrace{\star\star\ldots\star}^{(D)80}\ \overbrace{vv\ldots v}^{(E)62}\ \overbrace{\star\star\ldots\star}^{(F)60}\ |\ \overbrace{vv\ldots v}^{(G)202}$$

where as before $\star$ means that the symbol has been erased and v that the symbol has been correctly received. This is a situation in which we cannot recover the sequence because we either do not have enough clean memory in between the blocks of erasures (we need 100 clean symbols) or when we do have clean memory, the size of the error patterns surpasses the recovering rate per window. Therefore, the decoding algorithm for MDP convolutional codes needs to skip over these erasures, leading to the loss of this information. A $[202, 101]$ MDS block code would not be a better choice either since in a block of 202 symbols there would be more than 101 erasures making that block undecodable. ■

This example shows that even with enough clean memory between bursts of erasures, we cannot always decode if the bursts are too large relative to a given window. Let us imagine the folowing scenario. In the places where clean memory appears we change our decoding direction from left-to-right to right-to-left. Suppose that we could split the sequence into windows starting from the end, such that erasures are less accumulated in those windows, i.e., such that reading the patterns right-to-left would provide us with a distribution of erasures having an appropriate density per window to be recovered. Moreover, suppose that the code properties are such that inversion in the decoding direction is possible. Then, we would possibly increase the decoding capability leading to less information loss.

Convolutional codes consider the information as a complete sequence where z indicates in which instant did each coefficient arrive. The sliding window property allows convolutional codes to move along the sequences and frame different sizes of window providing them with a lot of flexibility. In order to apply a process where we can slide from right-to-left we will construct codes that are able to generate these

Universitat d'Alacant
Universidad de Alicante

sequences in inverted order, that is, from instant l to 0. Therefore, these codes will be able to execute a recovering process that starts at the end of the sequence and progresses up till the beginning. From now on we will refer to this inverted recovering process as *backward decoding* and to the normal left-to-right process as *forward decoding*.

We will show how this forward and backward flexibility of convolutional codes allow to recover patterns of erasures that block codes cannot recover. In order to do so we recall the following results.

Proposition 3.1 ([36], Proposition 2.9): *Let $\mathcal{C}$ be an (n,k,δ)-convolutional code with minimal generator matrix $G(z)$. Let $\overline{G}(z)$ be the matrix obtained by replacing each entry $g_{ij}(z)$ of $G(z)$ by $\overline{g_{ij}}(z) = z^{\delta_j} g_{ij}(z^{-1})$, where δ_j is the j-th column degree of $G(z)$. Then, $\overline{G}(z)$ is a minimal generator matrix of an (n,k,δ)-convolutional code $\overline{\mathcal{C}}$, having the characterization*

$$\boldsymbol{v}_0 + \boldsymbol{v}_1 z + \cdots + \boldsymbol{v}_{s-1} z^{s-1} + \boldsymbol{v}_s z^s \in \mathcal{C}$$

if and only if

$$\boldsymbol{v}_s + \boldsymbol{v}_{s-1} z + \cdots + \boldsymbol{v}_1 z^{s-1} + \boldsymbol{v}_0 z^s \in \overline{\mathcal{C}}.$$

Universitat d'Alacant
Universidad de Alicante

We call $\overline{\mathcal{C}}$ the **reverse code** of $\mathcal{C}$. Similarly, we denote by $\overline{H}(z) = \sum_{i=0}^{\nu} \overline{H}_i z^i$ the parity check matrix of $\overline{\mathcal{C}}$. The reason for introducing $\overline{\mathcal{C}}$ is that it allows us to invert the time of our sequence allowing us to describe the backward process we are looking for.

Although $\mathcal{C}$ and $\overline{\mathcal{C}}$ have the same d_{free}, they may have different values for the column distances since the truncations of the code words $\mathbf{v}(z) = \sum_{i=0}^{s} \boldsymbol{v}_i z^i$ and $\overline{\mathbf{v}}(z) = \sum_{i=0}^{s} \boldsymbol{v}_{s-i} z^i$ do not involve the same coefficients:

$$\begin{aligned} d_j^c(\mathcal{C}) &= \min\left\{\mathrm{wt}(\mathbf{v}_{[0,j]}(z)) \mid \mathbf{v}(z) \in \mathcal{C} \quad \text{and} \quad \boldsymbol{v}_0 \neq \mathbf{0}\right\} \\ &= \min\left\{\sum_{i=0}^{j} \mathrm{wt}(\boldsymbol{v}_i) \mid \mathbf{v}(z) \in \mathcal{C} \quad \text{and} \quad \boldsymbol{v}_0 \neq \mathbf{0}\right\} \\ d_j^c(\overline{\mathcal{C}}) &= \min\left\{\mathrm{wt}(\overline{\mathbf{v}}_{[0,j]}(z)) \mid \overline{\mathbf{v}}(z) \in \overline{\mathcal{C}} \quad \text{and} \quad \overline{\boldsymbol{v}}_0 \neq \mathbf{0}\right\} \\ &= \min\left\{\sum_{i=0}^{s-j+1} \mathrm{wt}(\boldsymbol{v}_{s-i}) \mid \mathbf{v}(z) \in \mathcal{C} \text{ and } \boldsymbol{v}_s \neq \mathbf{0}\right\}. \end{aligned}$$

Similar to the forward decoding process, in order to achieve maximum recovering rate per window when recovering using backward decoding, we need the column distances of $\overline{\mathcal{C}}$ to be maximal up to a point. This leads to the following definition.

Definition 3.1: Let $\mathcal{C}$ be an MDP (n, k, δ)-convolutional code. We say that $\mathcal{C}$ is a reverse-MDP convolutional code if the reverse code $\overline{\mathcal{C}}$ of $\mathcal{C}$ is an MDP code as well.

As previously explained, reverse-MDP convolutional codes are better candidates than MDP convolutional codes for recovering over the erasure channel. We will prove that the existence of this class of codes is guaranteed over fields with enough number of elements. For this we recall that set of all convolutional codes from a quasi-projective variety [34]. This quasi-projective variety can be also seen as a Zariski open subset of the projective variety described in [67, 72]. In [39] it was shown that MDP codes form a generic set when viewed as a subset of the quasi-projective variety of all (n, k, δ)-convolutional codes. Following similar ideas to the ones in the proof of the existence of MDP convolutional codes [39] we will show that reverse-MDP codes form a nonempty Zariski open set of the quasi-projective variety of generic convolutional codes and moreover, that the components of the elements of this set are contained in a finite field or a finite extension of it.

Universitat d'Alacant
Universidad de Alicante

Remark 3.2: The proof for the existence of generic MDP convolutional codes given in [39] is based in the systems theory representation of convolutional codes introduced in Section 1.4, the (A, B, C, D) representation. Since this one is closely related with the submodule point of view that we consider and a convolutional code can be represented in either way we will use in our proof the same notions as in [39]. See [70, 71, 73] for further references and details on systems theory representations. In [39] the set of MDP convolutional codes is described by sets of matrices $\{F_0, F_1, \ldots, F_L\}$ that form a matrix $\mathcal{T}_L$ with the MDP property. The matrices $\{F_0, F_1, \ldots, F_L\}$ are directly related with the representation (A, B, C, D) and have its elements in $\bar{\mathbb{F}}$, the closure of certain finite base field $\mathbb{F}$. $\bar{\mathbb{F}}$ is an infinite field. Based on the minors that must be different from zero in $\mathcal{T}_L$ for $\mathcal{C}$ to be MDP a set of finitely many polynomial equations is given. This set describes the codes that do not hold the MDP property. The zeros of each of these polynomials describe a proper algebraic subset of $\bar{\mathbb{F}}^{(L+1)(n-k)k}$. The complement of these subsets are nonempty Zariski open sets in $\bar{\mathbb{F}}^{(L+1)(n-k)k}$ whose intersection is a nonempty Zariski open set because

there is a finite number of them. Thus, the set of MDP codes forms a nonempty Zariski open subset of the quasi-projective variety of convolutional codes. ■

Theorem 3.2: *Let k, n and δ be positive integers. A reverse-MDP (n, k, δ)-convolutional code exists over a sufficiently large field.*

PROOF: Let $\mathbb{F}$ be a finite field and $\bar{\mathbb{F}}$ be its algebraic closure. Following a similar reasoning as in [39] we will show that the set of reverse-MDP convolutional codes forms a generic set when viewed as a subset of the quasi-projective variety of all (n, k, δ)-convolutional codes since it is the intersection of two nonempty Zariski open sets: the one of MDP codes and the one of the codes whose reverse is MDP.

As proved in [39], there exist sets of finitely many polynomial equations whose zero sets describe those convolutional codes that are not MDP. Each of these sets is a proper subset of $\bar{\mathbb{F}}^{(L+1)(n-k)k}$ and its complement is a nonempty Zariski open set in $\bar{\mathbb{F}}^{(L+1)(n-k)k}$. Let $\{W_j\}_{j=0}^{\theta}$ denote those complements, where θ is a positive integer. With a similar set of finitely many polynomial equations one can describe those codes whose reverse ones are not MDP. These zero sets are proper algebraic sets over $\bar{\mathbb{F}}^{(L+1)(n-k)k}$ and the complements of those, for instance $\{U_j\}_{j=0}^{\phi}$, with ϕ a positive integer, are again nonempty Zariski open sets in $\bar{\mathbb{F}}^{(L+1)(n-k)k}$. Let V be the intersection of all of them

$$V = \left(\bigcap_{j=0}^{\theta} W_j\right) \bigcap \left(\bigcap_{j=0}^{\phi} U_j\right).$$

Thus V is a nonempty Zariski open set since there are finitely many sets in the intersection. V describes the set of reverse-MDP codes. If we take one element in V, i.e., we select the matrices $\{F_0, F_1, \ldots F_L\}$ that present a certain reverse-MDP code $\mathcal{C}$, then we have finitely many entries and either all of them belong to $\mathbb{F}$ or to a finite extension field of $\mathbb{F}$ that contains them all. Thus, choosing $\mathbb{F}$ or the extension we can always find a finite field where reverse-MDP codes exist. □

Remark 3.3: The equations characterizing the set of reverse-MDP convolutional codes can be made very explicit for codes of degree δ, where $(n-k) \mid \delta$ and the code has a parity check matrix $H(z) = H_0 + H_1 z + \cdots + H_\nu z^\nu$. In this case the reverse code has parity check matrix $\overline{H}(z) = H_\nu + H_{\nu-1} z + \cdots + H_0 z^\nu$ and $\overline{H}(z)$ is

reverse-MDP if and only if the conditions from Theorem 1.3 hold and in addition every full size minor of the matrix

$$\begin{bmatrix} H_\nu & H_{\nu-1} & \cdots & H_{\nu-L} \\ & H_\nu & \cdots & H_{\nu-L-1} \\ & & \ddots & \vdots \\ & & & H_\nu \end{bmatrix}$$

formed from the columns with indices $j_1, j_2, \ldots, j_{(L+1)(n-k)}$ with $j_{s(n-k)+1} > sn$ for $s = 1, 2, \ldots, L$, is nonzero. ■

Example 3.3: Let $\mathcal{C}$ be the $(2, 1, 1)$-convolutional code over $\mathbb{F} = GF(2^2)$ given by the parity check matrix

$$H(z) = \begin{bmatrix} 1 + \alpha^2 z & 1 + z \end{bmatrix}$$

where α satisfies $\alpha^2 + \alpha + 1 = 0$. We show that $\mathcal{C}$ is a reverse-MDP code. Observing matrix

$$\mathcal{H}_L = \begin{bmatrix} 1 & 1 & & \\ \alpha^2 & 1 & 1 & 1 \end{bmatrix}$$

one can see that all the 2×2 minors, except the one formed by columns 3 and 4, are non zero, so $\mathcal{C}$ is an MDP code. Moreover, the reverse code $\overline{\mathcal{C}}$ is defined by the matrix

$$\overline{H}(z) = \begin{bmatrix} z + \alpha^2 & z + 1 \end{bmatrix} = \overline{H}_0 + \overline{H}_1 z.$$

and

$$\overline{\mathcal{H}}_L = \begin{bmatrix} \alpha^2 & 1 & & \\ 1 & 1 & \alpha^2 & 1 \end{bmatrix}$$

where again only the minor formed by columns 3 and 4 is zero. This shows that $\overline{\mathcal{C}}$ is an MDP code and therefore $\mathcal{C}$ is a reverse-MDP convolutional code. ■

One can apply the backwards process directly to the received sequence moving

right-to-left, taking into account that

$$\begin{bmatrix} \overline{H}_\nu & \overline{H}_{\nu-1} & \cdots & \overline{H}_{\nu-j} & \cdots & \overline{H}_0 & & & \\ & \overline{H}_\nu & \cdots & \overline{H}_{\nu-j+1} & \cdots & \overline{H}_1 & \overline{H}_0 & & \\ & & \ddots & & & & & \ddots & \\ & & & \overline{H}_\nu & \cdots & \overline{H}_j & \overline{H}_{j-1} & \cdots & \overline{H}_0 \end{bmatrix} \begin{bmatrix} \boldsymbol{v}_{t+j} \\ \boldsymbol{v}_{t+j-1} \\ \vdots \\ \boldsymbol{v}_{t+1} \\ \boldsymbol{v}_t \end{bmatrix} = \mathbf{0} \quad (3.2)$$

if and only if

$$\begin{bmatrix} S_\nu & S_{\nu-1} & \cdots & S_{\nu-j} & \cdots & S_0 & & & \\ & S_\nu & \cdots & S_{\nu-j+1} & \cdots & S_1 & S_0 & & \\ & & \ddots & & & & & \ddots & \\ & & & S_\nu & \cdots & S_j & S_{j-1} & \cdots & S_0 \end{bmatrix} \begin{bmatrix} \overline{\boldsymbol{v}}_{t+j} \\ \overline{\boldsymbol{v}}_{t+j-1} \\ \vdots \\ \overline{\boldsymbol{v}}_{t+1} \\ \overline{\boldsymbol{v}}_t \end{bmatrix} = \mathbf{0} \quad (3.3)$$

where $S_i = \mathcal{J}_{(n-k)}\overline{H}_i\mathcal{J}_n$ for $i = 0, \ldots, \nu$, $\overline{\boldsymbol{v}}_j = \boldsymbol{v}_j\mathcal{J}_n$ for $j = 0, 1, 2, \ldots$ and $\mathcal{J}_r$ is an $r \times r$ matrix of the form

$$\mathcal{J}_r = \begin{bmatrix} 0 & 0 & \cdots & 0 & 1 \\ 0 & 0 & \cdots & 1 & 0 \\ \vdots & \vdots & & \vdots & \vdots \\ 0 & 1 & \cdots & 0 & 0 \\ 1 & 0 & \cdots & 0 & 0 \end{bmatrix} \quad \text{for} \quad r = 1, 2, \ldots, n.$$

Since both matrices are related by permutations of rows and columns, then the matrix in expression (3.2) satisfies the reverse-MDP property if and only if the matrix in expression (3.3) does. So one can work with the last one to recover the erasures without making any transformation on the received sequence.

We revisit the situation of Example 3.2 and show how reverse-MDP convolutional codes can recover the erasures that were rended undecodable by MDP convolutional codes.

Universitat d'Alacant
Universidad de Alicante

Example 3.4: Assume that the $(2, 1, 50)$- convolutional code that we use to transmit in example 3.2 is a reverse-MDP convolutional code. The reverse code $\overline{\mathcal{C}}$ has the same recovering rate per window as $\mathcal{C}$.

Recall that we were not able to recover the received sequence using a left-to-right process. We will do this by using a backward recovering.

Once we have received 100 symbols of C we can recover part of the past erasures. If we take the following window

$$\overbrace{v v \star \star v v \star \star \ldots v v \star \star}^{(B)180} \,|\, \overbrace{v v \ldots v}^{(C)100}$$

and we use the reverse code $\overline{\mathcal{C}}$ to solve the inverted system then we can recover the erasures in B. Moreover, taking 100 clean symbols from G, the 60 erasures in F and 60 more clean symbols from E

$$\overbrace{v v \ldots v}^{(E)60} \overbrace{\star \star \ldots \star}^{(F)60} \,|\, \overbrace{v v \ldots v}^{(G)100}$$

we can in the same way recover block F. Like this we recovered 150 erasures that is more than 59% of the erasures happened in that concrete part of the sequence.■

In the previous example we showed how reverse-MDP convolutional codes and the backward process make possible to recover information that would already be considered as lost by an MDS block code, because it is happening in a previous block, or by an MDP code, because it cannot move backward. We use a space of clean memory, not only to recover the next bust of erasures, but additionally to recover the previous one. We can do this as soon as we received enough clean symbols and we do not need to wait until we receive a whole new block.

If we would allow this backward process to be complete, that is, to go from the end of the sequence up to the beginning, we would recover much more information. In this case we do not consider this situation since it would imply that we need to wait until the whole sequence was received in order to start recovering right-to-left and that would not give better results than the retransmission of lost packets.

In Algorithm 1 we present the recovering algorithm for a sequence of length l. The value 0 represents a packet that has not been received; 1 represents a correctly received packet; $\mathbf{1}$, a vector of ones, represents a clean window; $\mathsf{findzeros}(\boldsymbol{v})$ is a function that returns a vector with the positions of the zeros in $\boldsymbol{v}$, and $\mathsf{forward}(\mathcal{C}, j, \boldsymbol{v})$

Algorithm 1: Recovering Algorithm

Data: $[\boldsymbol{v}_0, \boldsymbol{v}_1, \ldots, \boldsymbol{v}_l]$, the received sequence.
Result: $[\boldsymbol{v}_0, \boldsymbol{v}_1, \ldots, \boldsymbol{v}_l]$, the corrected sequence.

```
begin
    i = 0
    while i ≤ l do
        forwardsucces = 0
        backwardsucces = 0
        if v_i = 0 then
            if [v_(i−νn), ..., v_{i−1}] = 1 then
                j = L
                while forwardsucces = 0 and j ≥ 0 do
                    if length(findzeros([v_i, ..., v_{i+(j+1)n−1}])) ≤ (j + 1)(n − k) then
                        [v_{i−νn}, ..., v_{i+(j+1)n−1}] = forward(C, j, [v_{i−νn}, ..., v_{i+(j+1)n−1}])
                        [v_{i−νn}, ..., v_{i+(j+1)n−1}] = forward(C, j, [v_{i−νn}, ..., v_{i+(j+1)n−1}])
                        forwardsucces = 1
                        i = i + (j + 1)n − 1
                    end
                    j = j − 1
                end
                if forwardsucces ≠ 1 then
                    k = i
                    while backwardsucces = 0 and k ≤ l do
                        if [v_k, ..., v_{k+νn−1}] = 1 then
                            j = L
                            while backwardsucces = 0 and j ≥ 0 do
                                if length(findzeros([v_{k−(j+1)n}, ..., v_{k−1}])) ≤ (j + 1)(n − k)
                                then
                                    [v_{k−(j+1)n}, ..., v_{k+νn−1}] =
                                    backward(C̄, j, [v_{k−(j+1)n}, ..., v_{k+νn−1}])
                                    backwardsucces = 1
                                    i = k + νn − 1
                                end
                                j = j − 1
                            end
                        end
                        k = k + 1
                    end
                end
            end
            i = i + 1
        end
    end
end
```

Universitat d'Alacant
Universidad de Alicante

and $\mathsf{backward}(\overline{\mathcal{C}}, j, \boldsymbol{v})$ are the forward and backward recovering functions, respectively. They use the parity check matrices of $\mathcal{C}$ and $\overline{\mathcal{C}}$ to recover the erasures that happen in $\boldsymbol{v}$ within a window of size $(j+1)n$.

Remark 3.4: Notice that the first and the last blocks of length $(j+1)n$ of the sequence (when using $\mathcal{C}$ and $\overline{\mathcal{C}}$, respectively) do not need the use of previous clean memory since we assume that $\boldsymbol{v}_i = \mathbf{0}$ for $i < 0$ and $i > l$, so we can solve the following systems

$$\begin{bmatrix} H_0 & & & \\ H_1 & H_0 & & \\ \vdots & \vdots & \ddots & \\ H_j & H_{j-1} & \cdots & H_0 \end{bmatrix} \begin{bmatrix} \boldsymbol{v}_0 \\ \boldsymbol{v}_1 \\ \vdots \\ \boldsymbol{v}_j \end{bmatrix} = \mathbf{0}, \quad j = 0, 1, \ldots, L,$$

$$\begin{bmatrix} H_L & \cdots & H_{L-j+1} & H_{L-j} \\ & \ddots & \vdots & \vdots \\ & & H_L & H_{L-1} \\ & & & H_L \end{bmatrix} \begin{bmatrix} \boldsymbol{v}_{l-j} \\ \boldsymbol{v}_{l-j-1} \\ \vdots \\ \boldsymbol{v}_l \end{bmatrix} = \mathbf{0}, \ j = 0, 1, \ldots, L.$$

■

Universitat d'Alacant
Universidad de Alicante

3.4 Construction of reverse-MDP convolutional codes

As we showed previously, reverse-MDP convolutional codes exist over sufficiently large fields and they give a good performance when decoding over the erasure channel. Unfortunately we do not have a general construction for this type of codes because for certain values of the parameters we do not know which is the relation between the matrices H_i and the matrices $\overline{H}_i$, $i = 0, 1, \ldots, \nu$. In this section we construct reverse-MDP codes for the case when $(n-k) \mid \delta$ and $k > \delta$ —when computing the parity check matrix— (or $k \mid \delta$ and $(n-k) > \delta$ —when constructing the generator matrix—).

Since reverse-MDP codes are codes satisfying both the forward and the backward MDP property, we could try to modify MDP convolutional codes such that

the corresponding reverse codes are also MDP. Recall from [37] that in the construction of MDP covolutional codes superregular matrices play an essential role. The construction in [37] is based on the use of superregular matrices (see, Chapter 2) in such a way that the nonzero minors of any size of a superregular matrix translate into nonzero full size minors of the parity check matrix of the code. This property characterized MDP convolutional codes (see, Theorem 1.3).

Motivated on this idea we propose a construction for reverse-MDP convolutional codes based on the use of reverse-superregular matrices (see, Chapter 2). These are specific matrices that are superregular forward and backward. The minors of these matrices will translate into full size minors of the parity check matrices of $\mathcal{C}$ and $\overline{\mathcal{C}}$.

Due to the importance that these matrices have in our construction we refer the reader to Chapter 2 where we provide with several tools in order to generate them.

Once we have generated the necessary matrices we can proceed to construct our codes. If we let $(n-k) \mid \delta$ and $k > \delta$ and extract appropriate columns and rows from a reverse-superregular matrix we can obtain the parity check matrix of a reverse-MDP code $\mathcal{C}$, that is, $\mathcal{C}$ and $\overline{\mathcal{C}}$ satisfy the MDP property.

Theorem 3.3: *Let A be an $r \times r$ reverse-superregular matrix with $r = (L+1)(2n-k-1)$. For $j = 0, 1, \ldots, L$ let I_j be sets of row indices*

$$I_j = \{(j+1)n + j(n-k-1), (j+1)n + j(n-k-1) + 1, \ldots, (j+1)(2n-k-1)\}$$

and J_j the sets of column indices

$$J_j = \{jn + j(n-k-1) + 1, jn + j(n-k-1) + 2, \ldots, (j+1)n + j(n-k-1)\}$$

and let I and J be the union of these sets, that is,

$$I = \bigcup_{j=0}^{L} I_j \quad \text{and} \quad J = \bigcup_{j=0}^{L} J_j.$$

Let $\widetilde{A}$ be the $(L+1)(n-k) \times (L+1)n$ lower block triangular submatrix of A formed by intersecting the rows indexed by the members of I with the columns indexed by the members of J in the following way

$$\widetilde{A} = A_J^I.$$

Universitat d'Alacant
Universidad de Alicante

Then every $(L+1)(n-k) \times (L+1)(n-k)$ full size minor of $\widetilde{A}$ formed from the columns with indices $1 \leq i_1 < \cdots < i_{(L+1)(n-k)}$ where $i_{s(n-k)} \leq sn$ for $s = 1, 2, \ldots, L$, is nonzero. Moreover, the same property holds for $\widetilde{A}_{\text{rev}}$.

PROOF: The proof follows from the fact that every full size minor of $\widetilde{A}$ formed from the columns with indices $1 \leq i_1 < \cdots < i_{(L+1)(n-k)}$ where $i_{s(n-k)} \leq sn$ for $s = 1, 2, \ldots, L$, is a not trivially zero minor of the former reverse-superregular matrix A and, therefore, is nonzero. If we take $\widetilde{A}_{\text{rev}}$, the mentioned minors are related to proper submatrices of A_{rev}. Since A is a reverse-superregular matrix, the corresponding not trivially zero minors of A_{rev} are nonzero. □

The condition $(n-k) \mid \delta$ and $k > \delta$ ensures that $L = \nu = \frac{\delta}{(n-k)}$. In this way $H_\nu = H_L$ and all the matrices of the expansion of $H(z)$ appear in $\mathcal{H}_L$ and we can completely define $H(z)$.

Let μ_j be the maximum degree of all polynomials in the j-th row of $H(z)$ and let H_∞ be the matrix whose j-th row is formed by the coefficients of z^{μ_j} in the j-th row of $H(z)$. One can note that in general $H_\infty \neq H_\nu$, but since $(n-k) \mid \delta$, H_ν has full rank, then both matrices coincide. Therefore, $\overline{H}_i = H_{\nu-i}$ for $i = 0, \ldots, \nu$ and the expression for the parity check matrix of $\overline{\mathcal{C}}$ is $\overline{H}(z) = H_\nu + H_{\nu-1}z + \cdots + H_1 z^{\nu-1} + H_0 z^\nu$.

In this way, the blocks of $\widetilde{A}$ are the appropriate matrices H_i and we can construct $H(z)$ and $\overline{H}(z)$. We illustrate the process with some examples.

Example 3.5: We can construct the parity check matrix of a reverse-MDP $(3, 2, 1)$-convolutional code $\mathcal{C}$ over $\mathbb{F} = \mathbb{F}_{2^5}$ using a 6×6 reverse-superregular matrix. Assume $\alpha^5 + \alpha^2 + 1 = 0$ and we have the reverse-superregular matrix

$$P = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ \alpha^{15} & 1 & 0 & 0 & 0 & 0 \\ \alpha^{21} & \alpha^{15} & 1 & 0 & 0 & 0 \\ \alpha^{23} & \alpha^{21} & \alpha^{15} & 1 & 0 & 0 \\ \alpha^{21} & \alpha^{23} & \alpha^{21} & \alpha^{15} & 1 & 0 \\ \alpha^{10} & \alpha^{21} & \alpha^{23} & \alpha^{21} & \alpha^{15} & 1 \end{bmatrix}.$$

Universitat d'Alacant
Universidad de Alicante

Applying Theorem 3.3 we can extract the corresponding blocks and obtain the matrix

$$\mathcal{H}_L = \begin{bmatrix} H_0 & O \\ H_1 & H_0 \end{bmatrix} = \begin{bmatrix} \alpha^{21} & \alpha^{15} & 1 & 0 & 0 & 0 \\ \alpha^{10} & \alpha^{21} & \alpha^{23} & \alpha^{21} & \alpha^{15} & 1 \end{bmatrix}$$

leading to the parity-check matrix of $\mathcal{C}$

$$H(z) = \begin{bmatrix} \alpha^{21} + \alpha^{10} z & \alpha^{15} + \alpha^{21} z & 1 + \alpha^{23} z \end{bmatrix}.$$

The parity-check matrix $\overline{H}(z)$ for $\overline{\mathcal{C}}$, which is given by

$$\overline{H}(z) = \sum_{i=0}^{1} \overline{H}_i z^i = \sum_{i=0}^{1} H_{1-i} z^i,$$

is now

$$\overline{H}(z) = \begin{bmatrix} \alpha^{10} + \alpha^{21} z & \alpha^{21} + \alpha^{15} z & \alpha^{23} + z \end{bmatrix}.$$

The matrix

$$\overline{H}_L = \begin{bmatrix} \alpha^{10} & \alpha^{21} & \alpha^{23} & 0 & 0 & 0 \\ \alpha^{21} & \alpha^{15} & 1 & \alpha^{10} & \alpha^{21} & \alpha^{23} \end{bmatrix}$$

has equivalent properties to the ones of the matrix

$$\mathcal{S}_L = \begin{bmatrix} \alpha^{23} & \alpha^{21} & \alpha^{10} & 0 & 0 & 0 \\ 1 & \alpha^{15} & \alpha^{21} & \alpha^{23} & \alpha^{21} & \alpha^{10} \end{bmatrix}$$

which we would have obtained applying Theorem 3.3 to the matrix P_{rev}. ■

Example 3.6: We can use a 8×8 reverse-superregular matrix over $\mathbb{F} = \mathbb{F}_{2^7}$ to construct a reverse-MDP $(4, 3, 1)$-convolutional code in the following way. If we apply Theorem 3.3 to the matrix

$$Q = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \alpha^{12} & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ \alpha^{32} & \alpha^{12} & 1 & 0 & 0 & 0 & 0 & 0 \\ \alpha^{45} & \alpha^{32} & \alpha^{12} & 1 & 0 & 0 & 0 & 0 \\ \alpha^{48} & \alpha^{45} & \alpha^{32} & \alpha^{12} & 1 & 0 & 0 & 0 \\ \alpha^{41} & \alpha^{48} & \alpha^{45} & \alpha^{32} & \alpha^{12} & 1 & 0 & 0 \\ \alpha^{27} & \alpha^{41} & \alpha^{48} & \alpha^{45} & \alpha^{32} & \alpha^{12} & 1 & 0 \\ \alpha^{21} & \alpha^{27} & \alpha^{41} & \alpha^{48} & \alpha^{45} & \alpha^{32} & \alpha^{12} & 1 \end{bmatrix}$$

where $\alpha^7 + \alpha^6 + 1 = 0$, we obtain the matrix

$$\mathcal{H}_L = \begin{bmatrix} H_0 & O \\ H_1 & H_0 \end{bmatrix} = \begin{bmatrix} \alpha^{45} & \alpha^{32} & \alpha^{12} & 1 & 0 & 0 & 0 & 0 \\ \alpha^{21} & \alpha^{27} & \alpha^{41} & \alpha^{48} & \alpha^{45} & \alpha^{32} & \alpha^{12} & 1 \end{bmatrix}.$$

Then the parity check matrix of $\mathcal{C}$ is

$$H(z) = \begin{bmatrix} \alpha^{45} + \alpha^{21}z & \alpha^{32} + \alpha^{27}z & \alpha^{12} + \alpha^{41}z & 1 + \alpha^{48}z \end{bmatrix}$$

and the parity check matrix of $\overline{\mathcal{C}}$ is

$$\overline{H}(z) = \begin{bmatrix} \alpha^{21} + \alpha^{45}z & \alpha^{27} + \alpha^{32}z & \alpha^{41} + \alpha^{12}z & \alpha^{48} + z \end{bmatrix}. \qquad \blacksquare$$

The same kind of construction can be applied in order to obtain the generator matrix of a code. Transposing the reverse-superregular matrix and changing the sizes for the extraction of rows and columns we obtain the following theorem similar to Theorem 3.3.

Theorem 3.4: *Let B be the transpose of an $r \times r$ reverse-superregular matrix with $r = (L+1)(n+k-1)$. For $j = 0, 1, \ldots, L$ let I_j be sets of row indices*

$$I_j = \{jn + j(k-1) + 1, jn + j(k-1) + 2, \ldots, (j+1)n + j(k-1)\}$$

and J_j the sets of column indices

$$J_j = \{(j+1)n + j(k-1), (j+1)n + j(k-1) + 1, \ldots, (j+1)(n+k-1)\}$$

and let I and J be the union of these sets

$$I = \bigcup_{j=0}^{L} I_j, \qquad J = \bigcup_{j=0}^{L} J_j.$$

Let $\widetilde{B}$ be the $(L+1)n \times (L+1)k$ upper block triangular submatrix of B formed by intersecting the rows indexed by the members of I with the columns indexed by the members of J in the following way

$$\widetilde{B} = B_J^I.$$

Then every $(L+1)k \times (L+1)k$ full size minor of $\widetilde{B}$ formed from the columns with indices $1 \leq i_1 < \cdots < i_{(L+1)k}$ where $i_{sk+1} > sn$ for $s = 1, \ldots, L$, is nonzero. Moreover, the same property holds for $\widetilde{B}_{\text{rev}}$.

Universitat d'Alacant
Universidad de Alicante

PROOF: Every full size minor of $\widetilde{B}$ formed from the columns with indices $1 \leq i_1 < \cdots < i_{(L+1)k}$ where $i_{sk+1} > sn$ for $s = 1, \ldots, L$, is a not trivially zero minor of the former transposed reverse-superregular matrix B and, therefore, is nonzero. Taking $\widetilde{B}_{\text{rev}}$, those full size minors correspond to the not trivially zero minors of B_{rev}, which by assumption are nonzero. □

In this case we take $k \mid \delta$ and $(n-k) > \delta$ so that $L = m = \frac{\delta}{k}$. Then all the matrices in the expansion of matrix $G(z)$ appear in $\mathcal{G}_L$. Like this we can completely define $G(z)$. As in the parity check matrix case, in general $G_\infty \neq G_m$, but with $k \mid \delta$, G_m has full rank and $G_\infty = G_m$. Now $\overline{G}_i = G_{m-i}$ for $i = 0, \ldots, m$ and the generator matrix of $\overline{\mathcal{C}}$ looks like $\overline{G}(z) = G_m + G_{m-1}z + \cdots + G_1 z^{m-1} + G_0 z^m$.

Extracting the blocks from $\widetilde{B}$ we can construct the generator matrix of the code as shown in the example below.

Example 3.7: We can construct the generator matrix of a $(3, 1, 1)$-convolutional code over $\mathbb{F} = GF(2^5)$. For this we use the transpose of a 6×6 reverse-superregular matrix. Let $\alpha^5 + \alpha^4 + \alpha^3 + \alpha^2 + 1 = 0$. We can apply Theorem 3.4 to the matrix

$$S = \begin{bmatrix} 1 & \alpha^{19} & \alpha^{16} & \alpha^{20} & \alpha^5 & \alpha^{16} \\ 0 & 1 & \alpha^{19} & \alpha^{16} & \alpha^{20} & \alpha^5 \\ 0 & 0 & 1 & \alpha^{19} & \alpha^{16} & \alpha^{20} \\ 0 & 0 & 0 & 1 & \alpha^{19} & \alpha^{16} \\ 0 & 0 & 0 & 0 & 1 & \alpha^{19} \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

obtaining

$$\mathcal{G}_L = \begin{bmatrix} G_0 & G_1 \\ O & G_0 \end{bmatrix} = \begin{bmatrix} \alpha^{16} & \alpha^{16} \\ \alpha^{19} & \alpha^5 \\ 1 & \alpha^{20} \\ 0 & \alpha^{16} \\ 0 & \alpha^{19} \\ 0 & 1 \end{bmatrix}.$$

Universitat d'Alacant
Universidad de Alicante

The generator matrices of $\mathcal{C}$ and $\overline{\mathcal{C}}$ are

$$G(z) = \begin{bmatrix} \alpha^{16} + \alpha^{16} z \\ \alpha^{19} + \alpha^{5} z \\ 1 + \alpha^{20} z \end{bmatrix} \quad \text{and} \quad \overline{G}(z) = \begin{bmatrix} \alpha^{16} + \alpha^{16} z \\ \alpha^{5} + \alpha^{19} z \\ \alpha^{20} + z \end{bmatrix}. \qquad \blacksquare$$

3.5 Complete-MDP convolutional codes

We explained earlier how reverse-MDP codes can improve the recovering process in comparison to MDP codes of the same parameters. However, the waiting time is still not reduced; after getting lost in the middle of a sequence, even though we are able to move in any direction with our decoding, we still need to come to a complete clean window of νn symbols before being able to start again the decoding process.

In this section we introduce complete-MDP convolutional codes. These codes assume stronger conditions on the parity check matrix of the code which reduce the number of clean symbols per window that one needs to observe to go back to the recovering process. R_ω decreases only at that concrete step and the waiting time gets shorter avoiding the loss of more information.

From now on we make the simplified assumption that $(n-k)$ divides the degree δ of the code, and the code $\mathcal{C}$ has a parity check matrix $H(z) = H_0 + H_1 z + \cdots + H_\nu z^\nu$. In other words H_ν has full rank and $\delta = \nu(n-k)$. Recall the definition of $L = \left\lfloor \frac{\delta}{k} \right\rfloor + \nu$.

The following matrix

$$\begin{bmatrix} H_\nu & \cdots & H_0 & & & \\ & H_\nu & \cdots & H_0 & & \\ & & \ddots & & \ddots & \\ & & & H_\nu & \cdots & H_0 \end{bmatrix}$$

will play an important role in the following. For this reason, we will call it the *partial parity-check matrix* of the code. Then we have the following definition.

Definition 3.2: A rate $\frac{k}{n}$ convolutional code $\mathcal{C}$ with parity check matrix $H(z)$ as above is called a complete-MDP convolutional code if in the $(L+1)(n-k) \times (\nu + L + 1)n$ partial parity check matrix every full size minor which is not trivially zero, is nonzero.

Remark 3.5: A full size minor of the partial parity check matrix formed from the columns $j_1, j_2, \ldots, j_{(L+1)(n-k)}$ is not trivially zero if and only if for $s = 1, 2, \ldots, L+1$, the following conditions are satisfied

- $j_{s(n-k)+1} > sn$,
- $j_{s(n-k)} \leq sn + \nu n$. ■

Based on many small examples, like Example 3.8, we conjecture the existence of complete-MDP convolutional codes for every set of parameters.

Example 3.8: The following parity check matrix represents a complete-MDP $(3, 1, 1)$-convolutional code over $\mathbb{F} = \mathbb{F}_{2^7}$ with $\alpha^7 + \alpha^6 + \alpha^3 + \alpha + 1 = 0$.

$$H(z) = \begin{bmatrix} \alpha^{76} + \alpha^{77}z & \alpha^{62} + \alpha^{85}z & 1 + \alpha^{76}z \\ \alpha^{73} + \alpha^{37}z & \alpha^{76} + \alpha^{77}z & \alpha^{62} + \alpha^{85}z \end{bmatrix}.$$

Universitat d'Alacant
Universidad de Alicante

Note that in this case $n - k = 2$ does not divide $\delta = 1$. The partial parity check matrix holds the condition that all its full size minors that are not trivially zero, that is, the ones that do not include columns 1, 2 and 3 or 7, 8 and 9, are nonzero.

$$\begin{bmatrix} \alpha^{77} & \alpha^{85} & \alpha^{76} & \alpha^{76} & \alpha^{62} & 1 & & & \\ \alpha^{13} & \alpha^{77} & \alpha^{85} & \alpha^{73} & \alpha^{76} & \alpha^{82} & & & \\ & & & \alpha^{77} & \alpha^{85} & \alpha^{76} & \alpha^{76} & \alpha^{62} & 1 \\ & & & \alpha^{13} & \alpha^{77} & \alpha^{85} & \alpha^{73} & \alpha^{76} & \alpha^{82} \end{bmatrix}.$$

■

Lemma 3.1: *Every complete-MDP convolutional code is reverse-MDP.*

PROOF: It follows from the fact that the matrices

$$\begin{bmatrix} H_0 & & & \\ H_1 & H_0 & & \\ \vdots & & \ddots & \\ H_L & H_{L-1} & \cdots & H_0 \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} H_\nu & H_{\nu-1} & \cdots & H_{\nu-L} \\ & H_\nu & & H_{\nu-L+1} \\ & & \ddots & \vdots \\ & & & H_\nu \end{bmatrix}$$

are included in the partial parity check matrix of the code. The not trivially zero full size minors of these matrices are not trivially zero full size minors of the partial parity check matrix and by Definition 3.2 they are nonzero as well. Therefore, the code is reverse-MDP. □

Note that the converse is not true in general, as we can see in the following example.

Example 3.9: The following matrix represents the parity check matrix of a reverse-MDP $(3, 1, 1)$-convolutional code over $\mathbb{F} = \mathbb{F}_{2^7}$ where $\alpha^7+\alpha^6+\alpha^5+\alpha^4+\alpha^2+\alpha+1 = 0$.

$$H(z) = \begin{bmatrix} \alpha^{93} + \alpha^{49}z & \alpha^{19} + \alpha^{30}z & \alpha^{75} + \alpha^{35}z \\ \alpha^{61} + \alpha^{19}z & \alpha^{93} + \alpha^{49}z & \alpha^{19} + \alpha^{30}z \end{bmatrix}.$$

Universitat d'Alacant
Universidad de Alicante

The code does not hold the complete-MDP condition because columns number 1, 5, 6 and 7 of the partial parity check matrix

$$\begin{bmatrix} \alpha^{49} & \alpha^{30} & \alpha^{35} & \alpha^{93} & \alpha^{19} & \alpha^{75} & & & \\ \alpha^{19} & \alpha^{49} & \alpha^{30} & \alpha^{61} & \alpha^{93} & \alpha^{19} & & & \\ & & & \alpha^{49} & \alpha^{30} & \alpha^{35} & \alpha^{93} & \alpha^{19} & \alpha^{75} \\ & & & \alpha^{19} & \alpha^{49} & \alpha^{30} & \alpha^{61} & \alpha^{93} & \alpha^{19} \end{bmatrix}$$

form a zero minor. ■

The use of this class of codes over the erasure channel gives some significant improvement to the recovering process. When we receive a pattern of erasures that we are not able to recover, by using complete-MDP codes, we do not need to wait until a complete clean window is received. Since all the full size minors of the partial parity check matrix are non zero, these codes behave locally, in a given (appropriately

sized) window, like an MDS block code in that bigger window size. It suffices to have a window with a certain percentage of clean symbols to continue with the recovering process. In addition, thanks to the sliding window property of a convolutional code, we can place that window where we find it convenient, in contrast to a block code, for which the blocks are fixed. This fact makes complete-MDP codes more powerful than MDS block codes.

The next theorem expresses more clearly the advantage of complete-MDP convolutional codes, visible in situations where the decoder gets lost because of an accumulation of too many erasures.

Theorem 3.5: *Let $\mathcal{C}$ be a complete-MDP (n,k,δ)-convolutional code. If in a window of size $(\nu+L+1)n$ there are not more than $(L+1)(n-k)$ erasures, and if they are distributed in such a way that between position 1 and sn and between positions $(\nu+L+1)n$ and $(\nu+L+1)n-s(n-k)$, for $s=1,2,\ldots,L+1$, there are not more than $s(n-k)$, then full correction of all symbols in this interval will be possible.*

PROOF: Assume we receive a window of size $(\nu+L+1)n$ with erasures in the positions $\{i_1, i_2, \ldots, i_{(L+1)(n-k)}\}$, with $1 \leq i_1 < i_2 < \cdots < i_{(L+1)(n-k)} \leq (\nu+L+1)n$. By assumption, the erasures are not too accumulated at the beginning or at the end of the window, that is,

- $i_{s(n-k)+1} > sn$ and
- $i_{s(n-k)} \leq sn + \nu n$

for $s = 1, 2, \ldots, L+1$. Therefore, the columns from the partial parity check matrix with indices $\{i_1, i_2, \ldots, i_{(L+1)(n-k)}\}$ form a not trivially zero minor (see Remark 3.5). We can write the following equation

$$\begin{bmatrix} H_\nu & H_{\nu-1} & \cdots & H_0 & & & \\ & H_\nu & \cdots & H_1 & H_0 & & \\ & & \ddots & & & \ddots & \\ & & & H_\nu & H_{\nu-1} & \cdots & H_0 \end{bmatrix} \begin{bmatrix} \boldsymbol{v}_t \\ \star \\ \boldsymbol{v}_{t+2} \\ \vdots \\ \star \\ \boldsymbol{v}_{t+\nu+L+1} \\ \star \end{bmatrix} = \mathbf{0}.$$

We have then a system where the coefficient matrix is obtained from the partial parity check matrix extracting the columns $\{i_1, i_2, \ldots, i_{(L+1)(n-k)}\}$ and the rest of the columns together with the correctly received symbols provide the independent terms. Since $\mathcal{C}$ is a complete-MDP convolutional code, the coefficient matrix of the system is full rank and we can compute the solution, in other words, we can recover all the symbols in that interval. After this we can continue with the recovering process. □

Complete-MDP convolutional codes have maximum recovering rate per window at any instant of the process, forward and backward, since they behave like a reverse-MDP code. However, when we find a pattern of erasures that we cannot recover these codes can get back to the process as fast as possible. The recovering rate per window at that instant decreases from $R_\omega = \frac{(L+1)(n-k)}{(L+1)n}$ to $R_\omega = \frac{(L+1)(n-k)}{(L+1+\nu)n}$, since we need to observe a bigger amount of correct information. However, the waiting time becomes shorter avoiding gaps in the sequence since we don't need anymore to observe a completely clean window.

The following example points out the kind of situations that make these codes more powerful than MDS block codes.

Example 3.10: Suppose that we use a MDS $[75, 50]$ block code to transmit a sequence over an erasure channel. This code has $R_\omega = \frac{25}{75}$. Assume that we are not able to recover the previous blocks of the sequence and let the following be the pattern received immediately after

$$\ldots \star\star \mid \overbrace{\star\star\ldots\star}^{(A)14}\overbrace{vv\ldots v}^{(B)21}\overbrace{\star\star\ldots\star}^{(C)12}\overbrace{vv\ldots v}^{(D)28} \mid$$

$$\mid \overbrace{vv\ldots v}^{(E)19}\overbrace{\star\star\ldots\star}^{(F)13}\overbrace{vv\ldots v}^{(G)30}\overbrace{\star\star\ldots\star}^{(H)13} \mid$$

$$\mid \overbrace{vv\ldots v}^{(I)30}\overbrace{\star\star\ldots\star}^{(J)6}\overbrace{vv\ldots v}^{(K)17}\overbrace{\star\star\ldots\star}^{(L)22} \mid \star\star\ldots$$

In this case the block code can not recover any of the erasures happened missing 80 information symbols.

Note that if we use an MDP or a reverse-MDP convolutional code with parameters $(3, 2, 16)$, in order to have the same recovering capability per window, we would neither be able to recover the erasures since one cannot find enough clean memory in between the bursts, that is, a space with at least 48 clean symbols.

Assume now that we use a complete-MDP $(3, 2, 16)$-convolutional code. The maximum recovering rate per window of this code is $R_\omega = \frac{25}{75}$ and the following rate for smaller window sizes $\frac{(j+1)(n-k)}{(j+1)n}$, $j = 0, 1, \ldots, 23$. We know as well that due to the complete-MDP property, if we get lost in the process we can start recovering again once we find a window of size $(L + 1 + \nu)n = (24 + 1 + 16)3 = 123$ where not more than 25 erasures occur.

For the above pattern, a possible such window is the following

$$\overbrace{vv\ldots v}^{(B)21}\overbrace{\star\star\ldots\star}^{(C)12}\overbrace{vv\ldots v}^{(D)28}\,|\,\overbrace{vv\ldots v}^{(E)19}\overbrace{\star\star\ldots\star}^{(F)13}\overbrace{vv\ldots v}^{(G)30}.$$

Once we have recovered this part we can go on with the next one

$$vv\ldots v\overbrace{\star\star\ldots\star}^{(H)13}\,|\,\overbrace{vv\ldots v}^{(I)30}$$

and finally recover

$$vv\ldots v\overbrace{\star\star\ldots\star}^{(J)6}\,|\,\overbrace{vv\ldots v}^{(K)17}.$$

Although we cannot recover block A with 14 erasures and block L with 22 we were able to recover more than 50% of the erasures ocurred in that part of the sequence, which is better than what an MDS block code could recover. ■

After this detailed analysis we present the results obtained from simulations. For these experiments we worked over erasure channels with increasing erasure probability. As we mentioned in Section 3.1 the probability that an erasure occurs after a first erasure has occurred increases, so if we denote by $P_{c|e}$ the probability that an erasure occurs after a correctly received symbol, and $P_{e|e}$ the probability that an erasure occurs after another erasure has already happened, then the following table shows examples of values taken for the simulations.

$P_{c\|e}$	0.16	0.22	0.34	0.4
$P_{e\|e}$	0.29	0.4	0.48	0.49

Figure 3.3 reflects the behavior of MDS codes over the erasure channel when choosing codes with different rates and over channels with different erasure probabilities. The recovering capability is expressed in terms of $\Phi = \frac{\#\text{erasures recovered}}{\#\text{erasures occurred}}$.

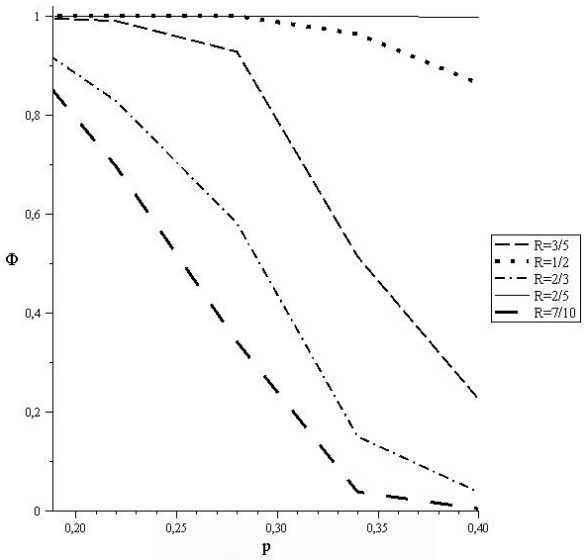
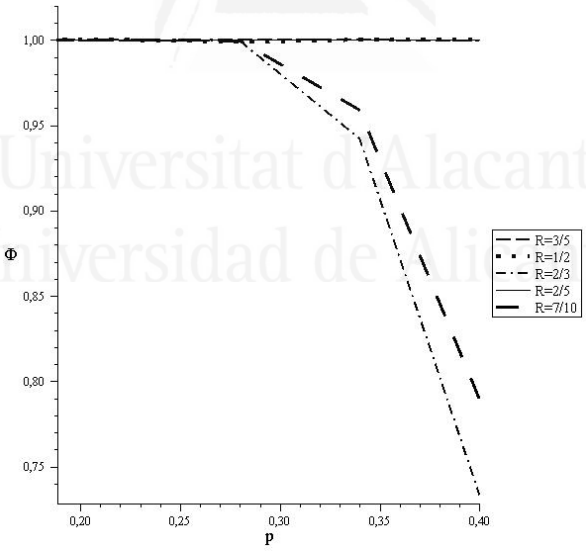


Figure 3.3: *Recovering capability (Φ) of MDS block codes with different rates in terms of the erasure probability of the channel (p).*



Universitat d'Alacant
Universidad de Alicante

Figure 3.4: *Recovering capability (Φ) of complete-MDP convolutional codes with different rates in terms of the erasure probability of the channel (p).*

In Figure 3.4 we can see the performance of complete-MDP convolutional codes. The codes were chosen to have equal transmission rate and recovering rate per window to the ones of the MDS block codes used in the simulations of Figure 3.3.

Observing the results one can see how complete-MDP convolutional codes give much better performance than MDS block codes. Even though the rate decreases for convolutional codes when we increase the erasure probability the behavior is better than in the MDS block case.

For this reason we propose this kind of codes as a very good alternative to MDS block codes over this channel. Moreover, we believe that our proposed way of generating reverse-superregular matrices in Theorem 2.4 together with the construction for reverse-MDP convolutional codes given in Section 3.4 generates complete-MDP convolutional codes since so far we did not find any evidence of the opposite. Unfortunately we were not able yet to prove this conjecture and this remains as an open question.

3.6 Conclusions

In this chapter, we propose reverse-MDP convolutional codes as an alternative to MDS block codes when decoding over an erasure channel. We have seen that the step-by-step-MDS property of the MDP codes doubled by the reverse MDP capacity of reverse-MDP convolutional codes, lets us recover a greater number of erasures in situations in which MDS block codes will skip a whole sequence. Even over large field sizes, the complexity of decoding is polynomial for a fixed window size, since the decoding algorithm requires only the solving of some linear systems. Moreover, the sliding window property of convolutional codes allows us to adapt the decoding process to the distribution of the erasures in the sequence. We have shown how the possibility of taking smaller windows lets us recover erasures that MDS block codes cannot recover.

We prove the existence of reverse-MDP convolutional codes and give an attempt on their construction. These codes give a better performance than MDP codes when working over the erasure channel.

In the last section we add stronger conditions to reverse-MDP convolutional codes to obtain complete-MDP convolutional codes. We show how, by taking advantage of the more powerful properties of these codes one can avoid getting stuck in the recovering process reducing the waiting time. Simulations show that complete-MDP convolutional codes perform better than MDS block, which makes the first an

attractive alternative to the latter. The properties of this new class of codes point them out as good candidates to achieve efficient algebraic decoding for convolutional codes.

Universitat d'Alacant
Universidad de Alicante

Chapter 4

The erasure channel and the system theory representation

4.1 Introduction

The achievement of an efficient decoding of convolutional codes under the appearance of errors is still an open problem. Viterbi's algorithm [27, 83, 89], list decoding [8, 92] or iterative methods [31, 40] have been implemented and even some algebraic decoding procedures [70] have been proposed, but it is still not clear how this should be done.

Universitat d'Alacant
Universidad de Alicante

On the other hand, as shown in Chapter 3, an interesting case to study is that of when only erasures occur. In this chapter we focus on the performance of convolutional codes over the erasure channel and we consider now the input-state-output representation given in Section 1.4. Convolutional codes are very powerful and adaptable when using this description as well.

Corollary 1.1 provides a main tool to recover a maximum amount of erasures when we use an MDP convolutional code to encode the information. In Section 4.2 we give an equivalent proof for Corollary 3.1.

In order to avoid the interruption of the recovering process for a long period of time and, therefore, the loss of big amounts of information, we provide three different possibilities to recalculate the state of the system. Firstly, we explain how the observability property helps us to recover $\boldsymbol{x}_t$ if too many erasures occur. Like that we can proceed with the decoding of our message when we observe a window of correctly received symbols. As a second option, if the observability matrix of the

system coincides with the parity check matrix of an MDS block code this result gets improved. To conclude, we impose conditions on the intersections of the column spaces generated by the matrices of the input-state-output representation. Thanks to these new assumptions, we can recover the state and part of the sequence at the same time after a big burst of erasures occurs.

4.2 Recovering a sequence over the erasure channel

In this section we use the mentioned properties of MDP convolutional codes to recover the maximum number of nonreceived symbols when transmitting over an erasure channel. We study this situation using the input-state-output representation of the code. We give an alternative proof of Corollary 3.1 in terms of this description.

Remark 4.1: Note that a minor formed by the columns with indices

$$\left\{i_1, i_2, \ldots, i_{(L+1)(n-k)}\right\}$$

from matrix

$$\left[\begin{array}{c|c} -I & \mathcal{F}_L \end{array}\right] = \left[\begin{array}{c|ccccc} & D & & & & \\ & CB & D & & & \\ -I & CAB & CB & D & & \\ & \vdots & \vdots & \vdots & \ddots & \\ & CA^{L-1}B & CA^{L-2}B & CA^{L-3}B & \cdots & D \end{array}\right]$$

is not trivially zero if and only if when we select the same minor out of the reordered matrix

$$\left[\begin{array}{cccccccccc} D & -I & & & & & & & & \\ CB & O & D & -I & & & & & & \\ CAB & O & CB & O & D & -I & & & & \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & & & \\ CA^{L-1}B & O & CA^{L-2}B & O & CA^{L-3}B & O & \cdots & D & -I \end{array}\right]$$

(that is, we take exactly the same columns which now are in a different position)

then the new indices $\left\{i'_1, i'_2, \ldots, i'_{(L+1)(n-k)}\right\}$ satisfy that

$$i'_{s(n-k)} \leq sn, \quad \text{for} \quad s = 1, 2, \ldots, (L+1)(n-k). \qquad \blacksquare$$

PROOF (COROLLARY 3.1): Assume we corrected, or we received correctly, the information sequence up to instant $t-1$. Equations (1.11) and (1.12) show that knowing the information sequence is equivalent to knowing the state of the system at a given instant. Then we can assume that $\boldsymbol{x}_t$ is known and we can write

$$\begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^L \end{bmatrix} \boldsymbol{x}_t + \left[\begin{array}{c|cccccc} & D & & & & \\ & CB & D & & & \\ -I & CAB & CB & D & & \\ & \vdots & \vdots & \vdots & \ddots & \\ & CA^{L-1}B & CA^{L-2}B & CA^{L-3}B & \cdots & D \end{array}\right] \left[\begin{array}{c} \boldsymbol{y}_t \\ \star \\ \boldsymbol{y}_{t+2} \\ \star \\ \vdots \\ \star \\ \boldsymbol{y}_{t+L} \\ \hline \star \\ \boldsymbol{u}_{t+1} \\ \vdots \\ \star \\ \boldsymbol{u}_{t+L-1} \\ \star \end{array}\right] = \mathbf{0},$$

where $\star$ represents blocks that have some erased components. Based on the MDP property of the code, matrix $[-I \mid \mathcal{F}_L]$ holds that each of its $(L+1)(n-k) \times (L+1)(n-k)$ full size minors that are not trivially zero are nonzero (see Remark 4.1).

Let $E = \{j_1, j_2, \ldots, i_s\}$, with $s \leq (L+1)(n-k)$, be the set of indices that denote the positions where the erasures occur. We can extract the columns given by the indices in E from matrix $[-I \mid \mathcal{F}_L]$ to form the coefficient matrix of a new system. The independent terms are calculated using the rest of the columns. Since we assume that only erasures happen, then the system has a unique solution and we can easily calculate it. As explained in the proof given in Chapter 3, we retain the part of the solution that is unique, that is, the first block, we slide down another n symbols along the sequence to frame the next window of length $(L+1)n$ and repeat the process. $\square$

Universitat d'Alacant
Universidad de Alicante

Note that in the encoding process only information symbols affect the rest of the sequence, hence, in the recovering process it is not necessary to pay attention to nonreceived parity symbols $\boldsymbol{y}_i$ such that $i < t$. Observing the form of matrix $[-I \mid \mathcal{F}_L]$ we can see that previous parity symbols are not related with the rest of the equations in the system and they are neither necessary to compute the current state $\boldsymbol{x}_t$. Therefore, a window is framed starting at an instant when an erased information symbol occurs. This erasure is placed in order that its column of coefficients in the system correspond to one of the first k columns of matrix $\mathcal{F}_L$.

As stated in Chapter 1, the maximality of the L-th column distance of the code implies the maximality of all the previous ones. Thus, the window size in the process is not necessarily fixed and can be adapted to the distribution of erasures in the sequence, that is, we can frame smaller windows and solve smaller systems using $[-I \mid \mathcal{F}_j]$, for $j \leq L$. This fact allows to start the decoding before a complete block has been received showing again that convolutional codes have in this sense more flexibility than block codes (see, Chapter 3).

4.2.1 State recovery

When too many erasures happen at a certain point in the sequence the systems presented in the previous section do not have a unique solution and we cannot recover the erased information. Since the knowledge of the sequence is equivalent to the knowledge of $\boldsymbol{x}_t$, we are not able to compute the state of the system after this point. As a consequence, the recovering process cannot continue and we can lose of information. Below we present three situations in which one can compute $\boldsymbol{x}_t$ under this kind of undesirable conditions. For this we use concrete properties of the code together with added assumptions that become more restrictive at each step. In the first case, we make use of the observability property of the code to recover $\boldsymbol{x}_t$ after observing a big enough amount of correctly received symbols. In a second situation, we let $\Omega_\xi(A, C)$ represent the parity check matrix of an MDS block code for a certain value of ξ. Then we have a better scenario and we can recover the state of the system observing a smaller clean window. We conclude the section with a more refined result. After some assumptions on the column subspaces of matrices $\Omega(A, C)$ and $[-I \mid \mathcal{F}_L]$ we recover simultaneously $\boldsymbol{x}_t$ and the erasures in the sequence.

4.2.1.1 Observability

In Section 1.4 we explained that if the pair (A, C) is observable one can compute $\boldsymbol{x}_t$ after observing for a certain period the input and the output of the system. The following result states exactly which is the amount of inputs and outputs that we need to observe until we are able to recover the state.

Theorem 4.1: *Let (A, B, C, D) represent an (n, k, δ)-convolutional code. If μ is the observability index of (A, C), then we can recalculate the state of the system when we observe a clean window of length $(\mu + 1)n$.*

PROOF: Assume a big burst of erasures not possible to be recovered occurred at instant $t_0 < t$. If we slide down along the sequence, or in other words, if at instant t we receive blocks

$$\begin{bmatrix} \boldsymbol{y}_t \\ \boldsymbol{u}_t \end{bmatrix}, \begin{bmatrix} \boldsymbol{y}_{t+1} \\ \boldsymbol{u}_{t+1} \end{bmatrix}, \ldots, \begin{bmatrix} \boldsymbol{y}_{t+\mu} \\ \boldsymbol{u}_{t+\mu} \end{bmatrix},$$

such that all the symbols have been correctly received, then we can write the following equation

$$\begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{\mu-1} \end{bmatrix} \begin{bmatrix} \star \\ \star \\ \vdots \\ \star \end{bmatrix}$$

$$+ \left[\begin{array}{c|cccccc} & D & & & & \\ & CB & D & & & \\ -I & CAB & CB & D & & \\ & \vdots & \vdots & \vdots & \ddots & \\ & CA^{\mu-2}B & CA^{\mu-3}B & CA^{\mu-4} & \cdots & D \end{array}\right] \left[\begin{array}{c} \boldsymbol{y}_t \\ \boldsymbol{y}_{t+1} \\ \vdots \\ \boldsymbol{y}_{t+\mu} \\ \hline \boldsymbol{u}_t \\ \boldsymbol{u}_{t+1} \\ \vdots \\ \boldsymbol{u}_{t+\mu} \end{array}\right] = \mathbf{0}, \qquad (4.1)$$

Universitat d'Alacant
Universidad de Alicante

where the $\star$-vector represents the *unkown* state of the system, $\boldsymbol{x}_t$. Since $\Omega_\mu(A, C)$ has full rank δ, there exists a combination of δ rows $\{j_1, j_2, \ldots, j_\delta\}$ from $\Omega_\mu(A, C)$ such that the associated $\delta \times \delta$ minor is nonzero. We restrict expression (4.1) to the equations corresponding with the row indices and we use the necessary blocks $\begin{bmatrix} \boldsymbol{y}_i \\ \boldsymbol{u}_i \end{bmatrix}$ to compute the independent terms. The resultant nonhomogeneous system with δ equations and δ variables has a unique solution. Therefore, we can calculate $\boldsymbol{x}_t$. Moreover, we can compute any state $\boldsymbol{x}_{t_s}$, for $t \leq t_s \leq t_r$, where t_r denotes the instant when the following erasure occurs. Using $\boldsymbol{x}_{t_r}$ we can go on with the recovering process, repeating the previous procedure as many times as needed. □

Notice that, similarly to the situation explained in Chapter 3, some information can be lost until we receive a clean window of size $(\mu + 1)n$. Theorems 4.2 and 4.3 add hypothesis on the input-state-output representation of $\mathcal{C}$ in order to reduce the amount of information symbols that we can lose. This minimizes the waiting time to continue with the recovering process.

4.2.1.2 MDS condition

In equation (4.1), the observability index μ ensures that $\operatorname{rank} \Omega_\mu(A, C) = \delta$, however, we do not know a priori which are the rows that form the equations of our system. Imposing some conditions on the observability matrix of the code we can control which are the rows which provide with a nonzero minor. In this way, the number of correct symbols that we need to observe in order to solve the system in expression (4.1) decreases.

Theorem 4.2: *If there exists an index ξ such that $\Omega_\xi(A, C)$ represents the transpose of the parity check matrix of an MDS block code, then we can recalculate the state of the system when we observe a clean window of length $\delta + \left\lceil \frac{\delta}{n-k} \right\rceil k$.*

PROOF: Assume that the recovering process was interrupted at instant $t_0 < t$ because a block with too many erasures was received. Let $\Omega_\xi(A, C)$ be the transpose of the parity check matrix of an MDS block code. Then any combination of δ rows

of the matrix has rank δ. We can restrict ourselves to the first δ equations

$$\overset{\delta}{\left\{\begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{\xi-1} \end{bmatrix}\right.} \begin{bmatrix} \star \\ \star \\ \vdots \\ \star \end{bmatrix}$$

$$+ \underbrace{\begin{bmatrix} \\ \\ -I_{\xi(n-k)} \\ \\ \\ \end{bmatrix}}_{\delta} \underbrace{\left|\begin{array}{ccccc} D & & & & \\ CB & D & & & \\ CAB & CB & D & & \\ \vdots & \vdots & \vdots & \ddots & \\ CA^{\xi-2}B & CA^{\xi-3}B & CA^{\xi-4}B & \cdots & D \end{array}\right.}_{\lceil \frac{\delta}{n-k} \rceil k} \left.\begin{array}{c} \\ \\ \\ \\ \\ \end{array}\right] \left[\begin{array}{c} \boldsymbol{y}_t \\ \boldsymbol{y}_{t+1} \\ \vdots \\ \boldsymbol{y}_{t+\xi} \\ \hline \boldsymbol{u}_t \\ \boldsymbol{u}_{t+1} \\ \vdots \\ \boldsymbol{u}_{t+\xi} \end{array}\right] = \mathbf{0},$$

Universitat d'Alacant
Universidad de Alicante

Given the dimensions of the matrices we need to observe $\lceil \frac{\delta}{n-k} \rceil k$ correct information symbols, $u_{i,j}$, together with its corresponding δ parity symbols, $y_{i,j}$ to compute the independent terms of the nonhomogeneous system with δ equations. Since the coefficient matrix is formed by the first δ rows of the parity check matrix of an MDS block code, is full rank. Therefore, we can recover $\boldsymbol{x}_t$. □

Thanks to the MDS condition of the observability matrix we are able to recover the state of the system using less clean symbols. This reduces the amount of missed information until we can proceed with the recovering process. In the following example we present a convolutional code holding the hypothesis in Theorem 4.2.

Example 4.1: Let $\mathbb{F} = \mathbb{F}_{2^r}$ and let the matrices

$$A = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad B, \quad C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \end{bmatrix} \text{ and } D,$$

with B and D any matrix. These matrices represent an $(n, k, 5)$-convolutional code with $n - k = 1$. Then $\Omega_6(A, C)$ represents the transpose of the parity check matrix of a repetition code. ■

4.2.1.3 Direct sum of subspaces

As presented above, the MDS assumption on the observability matrix of the system helps us to diminish the length of the clean window that we need to receive in order to recover $\boldsymbol{x}_t$. But this implies to wait until a certain number of contiguous nonerased symbols happen. Since not only bursts of erasures can occur, the waiting time for a clean block can become considerable. If we allow some erasures in a window we can earlier find appropriate scenarios to recover the state of the system.

Theorem 4.3 assumes stronger conditions on the subspaces generated by the matrices (A, B, C, D). As a result, $\boldsymbol{x}_t$ can be recovered when we find a window where not *too many* erasures occur. The nonreceived symbols are recovered and the process is faster retaken.

From now on, let $\alpha = (L+1)(n-k)-\delta$ and let $\chi \leq \binom{(L+1)n}{\alpha}$. Denote by $U_{j_1,j_2,\ldots,j_\alpha}$ the subspace generated by the columns with indices $\{j_1, j_2, \ldots, j_\alpha\}$ from the matrix $[-I \mid \mathcal{F}_L]$, where $1 \leq j_1 < j_2 < \cdots < j_\alpha \leq \chi$ and

$$\{j_1, j_2, \ldots, j_\alpha\} \subset \left\{i_1, i_2, \ldots, i_{(L+1)(n-k)}\right\},$$

with $i_{s(n-k)} \leq sn$ for $s = 1, 2, \ldots, (L+1)(n-k)$. In other words, the columns with indices $\{j_1, j_2, \ldots, j_\alpha\}$ are part of a not trivially zero minor of the matrix $[-I \mid \mathcal{F}_L]$ (see Remark 4.1). Then we can state the following result.

Theorem 4.3: *Let $V = \mathrm{colspa}(\Omega_{L+1}(A,C))$. If for all such subspaces $U_{j_1,j_2,\ldots,j_\alpha}$, the condition $U_{j_1,j_2,\ldots,j_\alpha} \oplus V = \mathbb{F}^{(L+1)(n-k)}$ is satisfied, then we can recalculate the state of the system when we observe a window of length $(L+1)n$ where not more than $(L+1)(n-k)-\delta$ erasures occur.*

PROOF: Assume that part of the transmitted information was not recovered due to the excessive number of erased symbols and we try to compute the state of the system at time t. For instance, let the following

$$\begin{bmatrix} \boldsymbol{y}_t \\ \star \end{bmatrix}, \begin{bmatrix} \star \\ \boldsymbol{u}_{t+1} \end{bmatrix}, \begin{bmatrix} \star \\ \star \end{bmatrix}, \ldots, \begin{bmatrix} \star \\ \boldsymbol{u}_{t+L-1} \end{bmatrix}, \begin{bmatrix} \star \\ \boldsymbol{u}_{t+L} \end{bmatrix},$$

be the part of sequence received at that instant, where $\star$ represents a block with some erasures. Note that the nonreceived symbols are not necessarily in adjacent positions. Let $\{j_1, j_2, \ldots, j_\alpha\}$ denote the positions of the erasures. We can write the following equation

$$\begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^L \end{bmatrix} \begin{bmatrix} \star \\ \star \\ \vdots \\ \star \end{bmatrix} + \left[\begin{array}{c|cccccc} & D & & & & \\ & CB & D & & & \\ -I & CAB & CB & D & & \\ & \vdots & \vdots & \vdots & \ddots & \\ & CA^{L-1}B & CA^{L-2}B & CA^{L-3}B & \cdots & D \end{array}\right] \left[\begin{array}{c} \boldsymbol{y}_t \\ \star \\ \star \\ \vdots \\ \star \\ \star \\ \hline \star \\ \boldsymbol{u}_{t+1} \\ \star\vdots \\ \boldsymbol{u}_{t+L-1} \\ \boldsymbol{u}_{t+L} \end{array}\right] = \mathbf{0}.$$

Therefore, if $U_{j_1,j_2,\ldots,j_\alpha} \oplus V = \mathbb{F}^{(L+1)(n-k)}$ then we can take the columns with indices $\{j_1, j_2, \ldots, j_\alpha\}$ from $[-I \mid \mathcal{F}_L]$ and the columns in $\Omega_{L+1}(A,C)$ to form the coefficient matrix of a new system. The rest of the columns in $[-I \mid \mathcal{F}_L]$ together

Universitat d'Alacant
Universidad de Alicante

with the correctly received symbols provide the independent terms of the system. The coefficient matrix of the system is full rank and we can compute the solution. This will determine the state of the system and the erased symbols in the sequence.

If $U_{j_1,j_2,\ldots,j_\alpha} \oplus V = \mathbb{F}^{(L+1)(n-k)}$, for all such subspaces $U_{j_1,j_2,\ldots,j_\alpha}$, then any pattern of erasures corresponding to the columns with indices $\{j_1, j_2, \ldots, j_\alpha\}$ can be recovered simultaneously with the state. Once we have correctly recovered $\boldsymbol{x}_t$ we can go on with the recovering process. □

The result presented in Theorem 4.3 allows us to recover the state of the system and the part of the sequence involved in that window at the same time. Moreover, it is not necessary anymore that the nonerased symbols appear together in the sequence, that is, we do not need to observe a complete clean block.

In order that the hypothesis of the theorem are satisfied we need that the columns in $\Omega_{L+1}(A, C)$ and any α columns of $[-I \mid \mathcal{F}_L]$ are linearly independent. If we choose our matrices in such a way that

$$\left[\begin{array}{c|c} \Omega_{L+1}(A,C) & \mathcal{F}_L \end{array}\right] = \begin{bmatrix} C & D & & & \\ CA & CB & D & & \\ \vdots & \vdots & \vdots & \ddots & \\ CA^L & CA^{L-1} & CA^{L-2} & \cdots & D \end{bmatrix}$$

is a proper submatrix of a superregular matrix, then the assumptions are already fulfilled. On the one hand, any combination of the columns in $\Omega_{L+1}(A, C)$ with α columns from $\mathcal{F}_L$ is directly a proper submatrix of the superregular matrix and therefore, the columns are linearly independent. On the other hand, any combination of columns which includes columns of the matrix $-I$ form a minor whose determinant can be reduced to the determinant of a proper submatrix of the superregular matrix, which again is nonzero. The details of how one should extract the columns and rows from a superregular matrix are presented in the theorem below.

Theorem 4.4: *Let M be an $r \times r$ superregular matrix with $r = \delta + (L+1)(n-1)$. For $j = 0, 1, \ldots, L$ let I_j be sets of row indices*

$$I_j = \{\delta + k + j(n-1), \delta + k + j(n-1) + 1, \ldots, \delta + (j+1)(n-1)\},$$

Universitat d'Alacant
Universidad de Alicante

and J_j the sets of column indices

$$J_0 = \{1, 2, \ldots, \delta + k\},$$
$$J_j = \{\delta + jn - (j-1), \delta + jn - (j-1) + 1, \ldots, \delta + k + j(n-1)\}$$
$$\text{for } j = 1, 2, \ldots, L,$$

and let I and J be the union of these sets

$$I = \bigcup_{j=0}^{L} I_j, \qquad J = \bigcup_{j=0}^{L} J_j.$$

Let $\widetilde{M}$ be the $(L+1)(n-k) \times \delta + (L+1)k$ lower block triangular submatrix of M formed by intersecting the rows indexed by the members of I with the columns indexed by the members of J in the following way

$$\widetilde{M} = M_J^I.$$

Then every $(L+1)(n-k) \times (L+1)(n-k)$ full size minor of $\widetilde{M}$, which contains the first δ columns, is nonzero.

Universitat d'Alacant
Universidad de Alicante

PROOF: The claim follows from the fact that the full size minors of $\widetilde{M}$ which contain the first δ columns correspond with proper submatrices of the superregular matrix M and therefore, are nonzero. □

In the example below we illustrate how one can construct a code satisfying the assumptions of Theorem 4.3.

Example 4.2: We construct a $(5, 3, 2)$-convolutional code. Then

$$L = \left\lfloor \frac{2}{3} \right\rfloor + \left\lfloor \frac{2}{2} \right\rfloor = 1.$$

Taking into account the dimensions of the matrices (A, B, C, D), we can use Theorem 4.4 to extract

$$\left[\begin{array}{c|cc} C & D & \\ CA & CB & D \end{array}\right]$$

from a 10×10 superregular matrix. Let $\mathbb{F} = \mathbb{F}_{397}$. We have the following 10×10 superregular matrix over $\mathbb{F}$

$$\left[\begin{array}{cccccccccc}
1 & & & & & & & & & \\
10 & 1 & & & & & & & & \\
15 & 10 & 1 & & & & & & & \\
130 & 15 & 10 & 1 & & & & & & \\
181 & 130 & 15 & 10 & 1 & & & & & \\
231 & 181 & 130 & 15 & 10 & 1 & & & & \\
345 & 231 & 181 & 130 & 15 & 10 & 1 & & & \\
83 & 345 & 231 & 181 & 130 & 15 & 10 & 1 & & \\
279 & 83 & 345 & 231 & 181 & 130 & 15 & 10 & 1 & \\
202 & 279 & 83 & 345 & 231 & 181 & 130 & 15 & 10 & 1
\end{array}\right].$$

Therefore, we have

$$\left[\begin{array}{c|cc} C & D & \\ CA & CB & D \end{array}\right] = \left[\begin{array}{cc|ccccc} 181 & 130 & 15 & 10 & 1 & & \\ 231 & 181 & 130 & 15 & 10 & & \\ 279 & 83 & 345 & 231 & 181 & 15 & 10 & 1 \\ 202 & 279 & 83 & 345 & 231 & 130 & 15 & 10 \end{array}\right].$$

Computing the corresponding realization (see, e.g., [5]) we obtain the (A, B, C, D) representation of our code

$$A = \left[\begin{array}{cc} 380 & 87 \\ 93 & 191 \end{array}\right], \; B = \left[\begin{array}{ccc} 272 & 237 & 1 \\ 24 & 307 & 0 \end{array}\right], \; C = \left[\begin{array}{cc} 181 & 130 \\ 231 & 181 \end{array}\right], \; D = \left[\begin{array}{ccc} 15 & 10 & 1 \\ 130 & 15 & 10 \end{array}\right].$$

One can check that the combination of any $(L+1)(n-k) - \delta = 2$ columns from

$$[-I \mid \mathcal{F}_L] = \left[\begin{array}{c|cccccc} & 15 & 10 & 1 & & & \\ & 130 & 15 & 10 & & & \\ -I & 345 & 231 & 181 & 15 & 10 & 1 \\ & 83 & 345 & 231 & 130 & 15 & 10 \end{array}\right]$$

together with the columns in

$$\Omega_{L+1}(A,C) = \begin{bmatrix} 181 & 130 \\ 231 & 181 \\ 279 & 83 \\ 202 & 279 \end{bmatrix}$$

results in a full rank matrix. In other words, the subspaces generated are linearly independent. ■

4.3 Conclusion

In this chapter we analyze the behavior of convolutional codes over the erasure channel focusing our interest on the input-state-output representation of these codes. As presented in Corollary 3.1, MDP codes achieve equal correcting capability to that of MDS block codes and turn out to be more flexible in many situations. We give a proof of this result using the (A, B, C, D) description in Section 4.2.

Like in Chapter 3, we study one of the main encountered problems: how to get back to the recovering process when this is interrupted. When too many erasures are received we cannot calculate the next state of the system. We propose three solutions that make possible the compute the state of the system, avoiding the loss of information. Each of these solutions makes use of certain characteristics of the code. The first one uses the observability property of the system. We show how we can recover $\boldsymbol{x}_t$ when we observe a certain amount of correct symbols. For the second one it is needed that the observability matrix of our system represents the parity check matrix of an MDS block code. As showed in Example 4.1, the MDS hypothesis is not a very restrictive assumption and one can find codes that satisfy this condition. In this case the number of necessary clean symbols decreases. The third solution requires stronger conditions on the subspaces generated by the matrices in order that we have a unique solution for our problem. Here some erasures per window are allowed and the waiting time is reduced. Both, the state and the erasures are recovered simultaneously. Even though the conditions are more restrictive, we explain how one can construct such codes using superregular matrices and we present in Example 4.2 a $(5, 3, 2)$-convolutional code fulfilling the hypothesis.

Universitat d'Alacant
Universidad de Alicante

Chapter 5

Periodically time-varying convolutional codes

5.1 Introduction

As we explained in Chapter 1 convolutional codes can be represented as time-invariant discrete linear systems over finite fields (see, e.g., [15, 42, 59]). However the idea of altering these systems and turning them into time-varying ones has interested many researchers and interesting results were already obtained. Thommesen and Justesen [80] realized that, for rates up to 4, time-varying codes of period 2 sufficed to get the best distance bounds. In [11], Costello proved a lower bound on the free distance of time-varying nonsystematic convolutional codes that is greater than the upper bound on d_{free} for time-invariant and time-varying systematic convolutional codes over all rates. This implies that nonsystematic time-varying convolutional codes exist which have a larger free distance than any time-invariant and time-varying systematic convolutional codes with comparable parameters. These results led to the study of its special properties and structure. In combination with wavelets [23] time-varying convolutional codes provided with unique trellis structures that resulted in fast and low computational complexity decoding algorithms; this type of time-varying codes can be decoded faster than comparable time-invariant convolutional codes. It was empirically shown that the number of good periodically time-varying convolutional codes increases exponentially with the period for any set of code parameters and some of them improve the free distance over fixed codes [51]. Mooser [60] showed that some periodically time-varying convolutional codes exist that result in decoded error rates that are less than those of any fixed

Universitat d'Alacant
Universidad de Alicante

code with comparable parameters. In addition, necessary and sufficient conditions for noncatastrophicity and computational efficient algorithms to test it have been developed using the polynomial representation of the codes [9, 61].

In this chapter we construct periodically time-varying convolutional codes using a first order representation. This particular construction maintain the degree δ of the system which helps not to increase decoding complexity. Moreover, the minimality of the (A, B, C, D) representation is guaranteed and therefore the noncatastropithity of the system. Examples show that the combination of nonoptimum subcodes can result in time-varying convolutional codes which have good distance characteristics, therefore we analyze its distance measures in the last section. We study the distance properties of the time-varying code and give a lower bound on the free distance. To follow the flow of the dissertation, where we have been mainly interested on the column distances of the code and the MDP property, we show that, even though the MDP conditions are the same for time-varying codes, the variability of the matrices in the different systems allows us to combine nonoptimal codes to obtain MDP ones. This fact points out time-varying codes as another tool to generate MDP codes.

5.2 Periodically time-varying convolutional codes

Universitat d'Alacant
Universidad de Alicante

In this section we define periodically time-varying convolutional codes and explain the concrete characteristics of our construction.

Let us assume that the matrices A_t, B_t, C_t and D_t at time t are of sizes $\delta \times \delta$, $\delta \times k$, $(n-k) \times \delta$ and $(n-k) \times k$, respectively. A **time-varying convolutional code** can be defined by means of the system

$$\left.\begin{array}{rcl} \boldsymbol{x}_{t+1} &=& A_t\boldsymbol{x}_t + B_t\boldsymbol{u}_t \\ \boldsymbol{y}_t &=& C_t\boldsymbol{x}_t + D_t\boldsymbol{u}_t \end{array}\right\}, \quad t = 0, 1, 2, \ldots, \quad \boldsymbol{x}_0 = \boldsymbol{0}. \tag{5.1}$$

If the matrices change periodically with periods τ_A, τ_B, τ_C and τ_D respectively, (that is, $A_{\tau_A+t} = A_t$, $B_{\tau_B+t} = B_t$, $C_{\tau_C+t} = C_t$ and $D_{\tau_D+t} = D_t$ for all t) then we have a **periodically time-varying convolutional code** of period $\tau = \operatorname{lcm}(\tau_A, \tau_B, \tau_C, \tau_D)$. For each fixed $t_0 \in \{0, 1, \ldots, \tau-1\}$ we say that the code

represented by $(A_{t_0}, B_{t_0}, C_{t_0}, D_{t_0})$ is a **subcode** of the time-varying convolutional code.

Any periodic time-varying convolutional code is equivalent to an invariant one (see, e.g., [9, 60, 61]). Relating every state and every output to previous states, we can always rewrite any block of τ iterations starting at a given time j as the system

$$\left.\begin{array}{rcl} X_{j+1} &=& \mathfrak{A}X_j + \mathfrak{B}U_j \\ Y_j &=& \mathfrak{C}X_j + \mathfrak{D}U_j \end{array}\right\} \tag{5.2}$$

where

$$\mathfrak{A} = \mathbb{A}_{\tau-1,0}, \quad \mathfrak{B} = \begin{bmatrix} \mathbb{A}_{\tau-1,1}B_0 & \mathbb{A}_{\tau-1,2}B_1 & \cdots & \mathbb{A}_{\tau-1,\tau-1}B_{\tau-2} & B_{\tau-1} \end{bmatrix},$$

$$\mathfrak{C} = \begin{bmatrix} C_0 \\ C_1\mathbb{A}_{0,0} \\ C_2\mathbb{A}_{1,0} \\ \vdots \\ C_{\tau-1}\mathbb{A}_{\tau-2,0} \end{bmatrix}, \quad \mathfrak{D} = \begin{bmatrix} D_0 & O & \cdots & O & O \\ C_1B_0 & D_1 & \cdots & O & O \\ C_2\mathbb{A}_{1,1}B_0 & C_2B_1 & \cdots & O & O \\ \vdots & \vdots & & \vdots & \vdots \\ C_{\tau-2}\mathbb{A}_{\tau-3,1}B_0 & C_{\tau-2}\mathbb{A}_{\tau-3,2}B_1 & \cdots & D_{\tau-2} & O \\ C_{\tau-1}\mathbb{A}_{\tau-2,1}B_0 & C_{\tau-1}\mathbb{A}_{\tau-2,2}B_1 & \cdots & C_{\tau-1}B_{\tau-2} & D_{\tau-1} \end{bmatrix},$$

$$X_j = \boldsymbol{x}_{\tau j}, \quad Y_j = \begin{bmatrix} \boldsymbol{y}_{\tau j} \\ \boldsymbol{y}_{\tau j+1} \\ \vdots \\ \boldsymbol{y}_{\tau j+\tau-1} \end{bmatrix}, \quad U_j = \begin{bmatrix} \boldsymbol{u}_{\tau j} \\ \boldsymbol{u}_{\tau j+1} \\ \vdots \\ \boldsymbol{u}_{\tau j+\tau-1} \end{bmatrix}$$

and

$$\mathbb{A}_{i,j} = \begin{cases} A_iA_{i-1}\cdots A_{j+1}A_j, & \text{if} \quad i \neq j, \\ A_i, & \text{if} \quad i = j. \end{cases}$$

System (5.2) is the time-invariant convolutional code equivalent to the periodic time-varying system (5.1). Our particular construction replaces A_t by a fixed matrix A and D_t by D for all t. Then, expression (5.1) turns into

$$\left.\begin{array}{rcl} \boldsymbol{x}_{t+1} &=& A\boldsymbol{x}_t + B_t\boldsymbol{u}_t \\ \boldsymbol{y}_t &=& C_t\boldsymbol{x}_t + D\boldsymbol{u}_t \end{array}\right\}, \quad t = 0, 1, 2, \ldots \quad \boldsymbol{x}_0 = \mathbf{0} \tag{5.3}$$

Universitat d'Alacant
Universidad de Alicante

and matrices $\mathfrak{A}$, $\mathfrak{B}$, $\mathfrak{C}$ and $\mathfrak{D}$ of system (5.2) become

$$\mathfrak{A} = A^{\tau}, \quad \mathfrak{B} = \begin{bmatrix} A^{\tau-1}B_0 & A^{\tau-2}B_1 & \cdots & AB_{\tau-2} & B_{\tau-1} \end{bmatrix},$$

$$\mathfrak{C} = \begin{bmatrix} C_0 \\ C_1A \\ \vdots \\ C_{\tau-1}A^{\tau-1} \end{bmatrix}, \quad \mathfrak{D} = \begin{bmatrix} D & O & \cdots & O & O \\ C_1B_0 & D & \cdots & O & O \\ C_2AB_0 & C_2B_1 & \cdots & O & O \\ \vdots & \vdots & & \vdots & \vdots \\ C_{\tau-2}A^{\tau-3}B_0 & C_{\tau-2}A^{\tau-4}B_1 & \cdots & D & O \\ C_{\tau-1}A^{\tau-2}B_0 & C_{\tau-1}A^{\tau-3}B_1 & \cdots & C_{\tau-1}B_{\tau-2} & D \end{bmatrix}.$$

5.2.1 Minimality conditions

In this subsection we study sufficient conditions to ensure the controllability and observability of the time-varying code. For this we use the equivalent time-invariant convolutional code obtained in (5.3). Note that Theorems 5.1 and 5.2 are constructive.

Theorem 5.1: *If $\tau k \geq \delta$ and the matrices B_j, for $j = 0, 1, \ldots, \tau-1$, are such that $B_j = A^{-(\tau-j-1)}E_j$, being $E_j \in \mathbb{F}^{\delta \times k}$ with $E = \begin{bmatrix} E_0 & E_1 & \cdots & E_{\tau-1} \end{bmatrix}$ a full rank matrix, then the system defined by expression (5.2) is controllable.*

PROOF: According to the form of the controllable matrix in (1.10) we have that

$$\Phi_{\delta}(\mathfrak{A}, \mathfrak{B}) = \begin{bmatrix} \mathfrak{B} & \mathfrak{A}\mathfrak{B} & \cdots & \mathfrak{A}^{\delta-1}\mathfrak{B} \end{bmatrix} = \begin{bmatrix} E & A^{\tau}E & \cdots & A^{(\delta-1)\tau}E \end{bmatrix}$$

which is clearly a full rank matrix. So, the system (5.2) is controllable. □

Similarly, we have the following result for the observability.

Theorem 5.2: *If $\tau(n-k) \geq \delta$ and the matrices C_j, for $j = 0, 1, \ldots, \tau-1$, are such that $C_j = F_jA^{-j}$, being $F_j \in \mathbb{F}^{(n-k)\times\delta}$ with $F = \begin{bmatrix} F_0' & F_1' & \cdots & F_{\tau-1}' \end{bmatrix}'$ a full rank matrix, then the system defined by expression (5.2) is observable.*

Note that if we choose B_j and C_j as in Theorems 5.1 and 5.2, then we ensure the minimality and the noncatastrophicity of the system.

Moreover, taking $A = I_\delta$, E as the parity check matrix of a block code, and F such that F_j is the submatrix of $\mathfrak{I} = \begin{bmatrix} I & I & I & \cdots \end{bmatrix}'$ formed by the rows $j(n-k)+l$ for $l = 1, 2, \ldots, n-k$, then we obtain the periodically time-varying convolutional code proposed in [62], in other words, our construction is a generalization of this concrete case.

Example 5.1: Let $\alpha \in \mathbb{F} = \mathbb{F}_{2^3}$ be a primitive element with $\alpha^3 + \alpha + 1 = 0$ and let

$$E = \begin{bmatrix} \alpha^5 & 1 & \alpha^5 \\ \alpha^5 & \alpha^5 & \alpha^3 \end{bmatrix}, \quad F = \begin{bmatrix} \alpha^4 & \alpha^5 \\ 1 & \alpha^4 \\ \alpha^3 & \alpha^5 \end{bmatrix}.$$

Following Theorems 5.1 and 5.2 we can obtain the $(2, 1, 2)$ subcodes

$$\mathcal{C}_0 = (A, B_0, C_0, D), \quad \mathcal{C}_1 = (A, B_1, C_1, D) \quad \text{and} \quad \mathcal{C}_2 = (A, B_2, C_2, D)$$

given by the matrices

$$A = \begin{bmatrix} \alpha^6 & \alpha \\ 0 & \alpha^2 \end{bmatrix}, \quad D = \begin{bmatrix} \alpha^6 \end{bmatrix},$$

$$B_0 = \begin{bmatrix} \alpha^5 \\ \alpha \end{bmatrix}, \quad B_1 = \begin{bmatrix} \alpha^6 \\ \alpha^3 \end{bmatrix}, \quad B_2 = \begin{bmatrix} \alpha^5 \\ \alpha^3 \end{bmatrix},$$

$$C_0 = \begin{bmatrix} \alpha^4 & \alpha^5 \end{bmatrix}, \quad C_1 = \begin{bmatrix} \alpha & \alpha^6 \end{bmatrix}, \quad C_2 = \begin{bmatrix} \alpha^5 & \alpha^4 \end{bmatrix}.$$

The time-varying convolutional code $\mathcal{C}$ of period $\tau = 3$ obtained from the combination of $\mathcal{C}_0$, $\mathcal{C}_1$ and $\mathcal{C}_2$ can be represented by means of the time-invariant equivalent system

$$\left.\begin{aligned} X_{j+1} &= \begin{bmatrix} \alpha^4 & \alpha^4 \\ 0 & \alpha^6 \end{bmatrix} X_j + \begin{bmatrix} \alpha^5 & 1 & \alpha^5 \\ \alpha^5 & \alpha^5 & \alpha^3 \end{bmatrix} U_j \\ Y_j &= \begin{bmatrix} \alpha^4 & \alpha^5 \\ 1 & \alpha^4 \\ \alpha^3 & \alpha^5 \end{bmatrix} X_j + \begin{bmatrix} \alpha^6 & 0 & 0 \\ \alpha^2 & \alpha^6 & 0 \\ \alpha^2 & \alpha^5 & \alpha^6 \end{bmatrix} U_j \end{aligned}\right\}.$$

Universitat d'Alacant
Universidad de Alicante

This is a minimal representation for $\mathcal{C}$.

Moreover, one can check that $\mathcal{C}_0$ and $\mathcal{C}_1$ are not MDS codes and only the free distance of $\mathcal{C}_2$ attains the generalized Singleton bound

$$d_{\text{free}}(\mathcal{C}_2) = d_5^c(\mathcal{C}_2) = 6 = (n-k)(\left\lfloor \frac{\delta}{k} \right\rfloor + 1) + \delta + 1.$$

However, the $(6, 3, 2)$ convolutional code $\mathcal{C}$ is an MDS code since $d_{\text{free}}(\mathcal{C}) = d_1^c(\mathcal{C}) = 6$.

Examples like this one show that the combination of codes whose d_{free} do not attain the maximum value can provide with codes which have a good distance behavior. In section 5.2.2 we study sufficient conditions on the subcodes to obtain optimum periodic time-varying convolutional codes in the sense of distance measures. ■

5.2.2 Distances

We dedicate this Subsection to study the distance measures of the time-varying convolutional codes contructed in this chapter.

If we recall the following result one can easily compute a lower bound on the free distance of the periodic time-varying code by means of the free distance of the invariant equivalent associated.

Universitat d'Alacant
Universidad de Alicante

Theorem 5.3 ([73], Theorem 3.1): *Let $\mathcal{C}$ be an observable (n, k, δ)-convolutional code defined through the matrices A, B, C and D. Let μ be the observability index of the pair (A, C) and suppose that there exists $d \in \mathbb{Z}^+$ such that $\Phi_{d\mu}(A, B)$ forms the parity-check matrix of a block code of distance d. Then $d_{\text{free}}(\mathcal{C}) \geq d$.*

Then we can state the next theorem.

Theorem 5.4: *If the matrix*

$$\Phi_{\delta+1}(\mathfrak{A}, \mathfrak{B}) = \begin{bmatrix} \mathfrak{B} & \mathfrak{A}\mathfrak{B} & \cdots & \mathfrak{A}^{(\delta+1)-2}\mathfrak{B} & \mathfrak{A}^{(\delta+1)-1}\mathfrak{B} \end{bmatrix}$$

represents the parity-check matrix of an MDS block code, then $d_{\text{free}}(\mathcal{C}) \geq \delta + 1$.

PROOF: Let $P = \begin{bmatrix} A^{\tau-1}B_0 & A^{\tau-2}B_1 & \cdots & B_{\tau-1} \end{bmatrix}$. Taking into account the minimality conditions given by Theorems 5.1 and 5.2 one can check that the observability index of our system is $\mu = 1$. The matrix

$$\begin{bmatrix} \mathfrak{B} & \mathfrak{A}\mathfrak{B} & \cdots & \mathfrak{A}^{((\delta+1)-1)}\mathfrak{B} \end{bmatrix} = \begin{bmatrix} P & A^{\tau}P & \cdots & A^{\delta\tau}P \end{bmatrix}$$

has size $\delta \times \tau k(\delta+1)$ and therefore the parameters of the corresponding block code are $[N, K] = [\tau k(\delta+1), \tau k(\delta+1) - \delta]$. The MDS hypothesis tells us that the minimum distance of the block code attains the Singleton bound which with the previous parameters equals $N - K + 1 = \delta + 1$. As a direct consequence of Theorem 5.3 we have $d_{\text{free}}(\mathcal{C}) \geq \delta + 1$. □

Concerning the column distances and MDP behavior of the time-varying convolutional codes we have constructed, we can say that the conditions for the maximality of the $d_j^c(\mathcal{C})$ and the MDP characterization remain as detailed in Section 1.4, when observing the code as the time-varying code itself and not as the equivalent time-invariant one. Notice that now equation (1.13) looks like

$$\begin{aligned} \mathcal{F}_j &= \begin{bmatrix} F_0 & & & \\ F_1 & F_0 & & \\ \vdots & & \ddots & \\ F_j & F_{j-1} & \cdots & F_0 \end{bmatrix} \\ &= \begin{bmatrix} D_0 & & & \\ C_1B_0 & D_1 & & \\ \vdots & & \ddots & \\ C_j\mathbb{A}^{j-2,1}B_0 & C_{j-1}\mathbb{A}^{j-2,2}B_1 & \cdots & D_{j-1} \end{bmatrix}. \end{aligned} \tag{5.4}$$

This change is crucial. Due to the fact that the matrices appearing in the products in F_i vary in each block we have more freedom when combining them. This means that only the products need to make $\mathcal{F}_j$ hold the nontrivial minor condition, but not that each of the former codes must satisfy it. In this way one has the possibility to combine nonMDP codes to obtain MDP ones. This situation is presented in Example 5.2. How one should choose the former nonMDP codes in order to obtain an MDP one is not yet clear. We make the discussion of the conditions for the case

Universitat d'Alacant
Universidad de Alicante

when the former subcodes are $(n, 1, 1)$-convolutional codes in the next subsection. The development for the cases with higher values of the parameters appear more complicated.

5.2.2.1 MDP conditions for the case $(n, 1, 1)$

For this study we assume that we work over $\mathbb{F}_p$ with $p \neq 2$. Let our time-varying code be formed by τ subcodes of parameters $(n, 1, 1)$. In this case we have

$$A_t = [a_t], \quad B_t = [b_t], \quad C_t = \begin{bmatrix} c_{t,1} \\ c_{t,2} \\ \vdots \\ c_{t,n-1} \end{bmatrix}, \quad D_t = \begin{bmatrix} d_{t,1} \\ d_{t,2} \\ \vdots \\ d_{t,n-1} \end{bmatrix},$$

where all the elements in C_t and D_t are different from 0 for $t = 0, 1, \ldots, \tau - 1$.

L attains the following values

$$L = \left\lfloor \frac{\delta}{k} \right\rfloor + \left\lfloor \frac{\delta}{n-k} \right\rfloor = \left\lfloor \frac{1}{1} \right\rfloor + \left\lfloor \frac{1}{n-1} \right\rfloor = \begin{cases} 1, & n > 2, \\ 2, & n = 2. \end{cases}$$

Then we need to check the maximality of $d_0^c(\mathcal{C})$ and $d_1^c(\mathcal{C})$ if $n > 2$ and $d_0^c(\mathcal{C})$, $d_1^c(\mathcal{C})$ and $d_2^c(\mathcal{C})$ if $n = 2$. If the column distances attain the upper bound then the code is MDP.

For the fist column distance we have $d_0^c(\mathcal{C}) \leq (n-1)(0+1)+1 = n$ and

$$\begin{aligned} d_0^c(\mathcal{C}) &= \min_{\boldsymbol{u}_0 \neq 0} \left\{ \sum_{i=0}^{0} \mathrm{wt}(\boldsymbol{u}_i) + \sum_{i=0}^{0} \mathrm{wt}(\boldsymbol{y}_i) \right\} \\ &= \min_{\boldsymbol{u}_0 \neq 0} \left\{ \mathrm{wt}(\boldsymbol{u}_0) + \mathrm{wt}(\boldsymbol{y}_0) \right\} \\ &= \min_{\boldsymbol{u}_0 \neq 0} \left\{ \mathrm{wt}(\boldsymbol{u}_0) + \mathrm{wt}(C_0 \boldsymbol{x}_0 + D_0 \boldsymbol{u}_0) \right\}. \end{aligned}$$

Since we take the minimum over $\boldsymbol{u}_0 \neq \boldsymbol{0}$, then $\mathrm{wt}(\boldsymbol{u}_0) = 1$. Since $\boldsymbol{x}_0 = \boldsymbol{0}$, then $C_0 \boldsymbol{x}_0 = \boldsymbol{0}$ and since all the elements in D_0 are nonzero, then $D_0 \boldsymbol{u}_0 \neq \boldsymbol{0}$ with $\mathrm{wt}(D_0 \boldsymbol{u}_0) = n - 1$. So we have $\mathrm{wt}(C_0 \boldsymbol{x}_0 + D_0 \boldsymbol{u}_0) = n - 1$ and hence $d_0^c(\mathcal{C}) = 1 + n - 1 = n$ reaches the upper bound.

Universitat d'Alacant
Universidad de Alicante

For the next column distance we have $d_1^c(\mathcal{C}) \leq (n-1)(1+1)+1 = 2n-1$ and

$$\begin{aligned} d_1^c(\mathcal{C}) &= \min_{\boldsymbol{u}_0 \neq \mathbf{0}} \left\{ \sum_{i=0}^{1} \mathrm{wt}(\boldsymbol{u}_i) + \sum_{i=0}^{1} \mathrm{wt}(\boldsymbol{y}_i) \right\} \\ &= \min_{\boldsymbol{u}_0 \neq \mathbf{0}} \left\{ \mathrm{wt}(\boldsymbol{u}_0) + \mathrm{wt}(\boldsymbol{u}_1) + \mathrm{wt}(\boldsymbol{y}_0) + \mathrm{wt}(\boldsymbol{y}_1) \right\} \\ &= \min_{\boldsymbol{u}_0 \neq \mathbf{0}} \left\{ \mathrm{wt}(\boldsymbol{u}_0) + \mathrm{wt}(\boldsymbol{u}_1) + \mathrm{wt}(C_0\boldsymbol{x}_0 + D_0\boldsymbol{u}_0) + \mathrm{wt}(C_1\boldsymbol{x}_1 + D_1\boldsymbol{u}_1) \right\}. \end{aligned}$$

Here $\boldsymbol{x}_1 = [a_0]\boldsymbol{x}_0 + [b_0]\boldsymbol{u}_0 = [b_0]\boldsymbol{u}_0 \neq \mathbf{0}$. The elements in C_1 are different from zero, then $C_1\boldsymbol{x}_1 \neq \mathbf{0}$ with weight $n-1$. However, $D_1\boldsymbol{u}_1$ can have weight 0 or $n-1$ depending on the value of $\boldsymbol{u}_1$. In the case the weight is 0 the bound is directly attained, but adding both terms $C_1\boldsymbol{x}_1 + D_1\boldsymbol{u}_1$ can provoke cancellations and a weight lower than desired. We study these 2 cases separately.

- $\boldsymbol{u}_0 \neq \mathbf{0}$ and $\boldsymbol{u}_1 = \mathbf{0}$.

$$\begin{aligned} &\mathrm{wt}(\boldsymbol{u}_0) + \mathrm{wt}(\boldsymbol{u}_1) + \mathrm{wt}(C_0\boldsymbol{x}_0 + D_0\boldsymbol{u}_0) + \mathrm{wt}(C_1\boldsymbol{x}_1 + D_1\boldsymbol{u}_1) \\ &= 1 + 0 + (n-1) + (n-1) = 2n-1. \end{aligned}$$

- $\boldsymbol{u}_0 \neq \mathbf{0}$ and $\boldsymbol{u}_1 \neq \mathbf{0}$. The column distance is maximized when the upper bound is attained in both cases. In order to reach the bound in this second case, we propose a concrete construction for the code in order that $d_1^c(\mathcal{C})$ is maximal.

$$\begin{aligned} &\mathrm{wt}(\boldsymbol{u}_0) + \mathrm{wt}(\boldsymbol{u}_1) + \mathrm{wt}(C_0\boldsymbol{x}_0 + D_0\boldsymbol{u}_0) + \mathrm{wt}(C_1\boldsymbol{x}_1 + D_1\boldsymbol{u}_1) \\ &= 1 + 1 + (n-1) + \mathrm{wt}(C_1\boldsymbol{x}_1 + D_1\boldsymbol{u}_1) \end{aligned}$$

For the column distance to be equal to $2n-1$ we need that the term $\mathrm{wt}(C_1\boldsymbol{x}_1 + D_1\boldsymbol{u}_1)$ satisfies

$$\mathrm{wt}(C_1\boldsymbol{x}_1 + D_1\boldsymbol{u}_1) = n-2.$$

In the sequel we discuss the necessary conditions to ensure this value.
We need that the following vector has at least $n-2$ components nonzero

$$\begin{bmatrix} c_{1,1} \\ c_{1,2} \\ \vdots \\ c_{1,n-1} \end{bmatrix} \boldsymbol{x}_1 + \begin{bmatrix} d_{1,1} \\ d_{1,2} \\ \vdots \\ d_{1,n-1} \end{bmatrix} \boldsymbol{u}_1 = [b_0] \begin{bmatrix} c_{1,1} \\ c_{1,2} \\ \vdots \\ c_{1,n-1} \end{bmatrix} \boldsymbol{u}_0 + \begin{bmatrix} d_{1,1} \\ d_{1,2} \\ \vdots \\ d_{1,n-1} \end{bmatrix} \boldsymbol{u}_1$$

Universidad de Alicante

If $\frac{d_{1,j}}{c_{1,j}} \neq \frac{d_{1,i}}{c_{1,i}}$ for all i, j only one element of the vector can be zero at a time and, therefore, we have a minimum weight of $n-2$. The column distance is now maximized and the code is MDP for $n > 2$.

When $n = 2$ we need to test $d_2^c(\mathcal{C})$ in addition. The upper bound is

$$d_2^c(\mathcal{C}) \leq (2-1)(2+1)+1 = 4.$$

Now $C_t = [c_t]$ and $D_t = [d_t]$. The column distance is now

$$\begin{aligned} d_2^c(\mathcal{C}) &= \min_{\boldsymbol{u}_0 \neq \mathbf{0}} \left\{ \sum_{i=0}^{2} \mathrm{wt}(\boldsymbol{u}_i) + \sum_{i=0}^{2} \mathrm{wt}(\boldsymbol{y}_i) \right\} \\ &= \min_{\boldsymbol{u}_0 \neq \mathbf{0}} \{ \mathrm{wt}(\boldsymbol{u}_0) + \mathrm{wt}(\boldsymbol{u}_1) + \mathrm{wt}(\boldsymbol{u}_2) + \mathrm{wt}(C_0\boldsymbol{x}_0 + D_0\boldsymbol{u}_0) \\ &\qquad + \mathrm{wt}(C_1\boldsymbol{x}_1 + D_1\boldsymbol{u}_1) \ + \mathrm{wt}(C_2\boldsymbol{x}_2 + D_2\boldsymbol{u}_2) \} . \end{aligned}$$

If $\boldsymbol{u}_1 \neq \mathbf{0}$ and $\boldsymbol{u}_2 \neq \mathbf{0}$ it can happen that $C_1\boldsymbol{x}_1 + D_1\boldsymbol{u}_1$ or $C_2\boldsymbol{x}_2 + D_2\boldsymbol{u}_2$ suffer cancellations and we obtain cases with lower weight. The following situations can occur.

- $\boldsymbol{u}_0 \neq \mathbf{0}$, $\boldsymbol{u}_1 = \mathbf{0}$ and $\boldsymbol{u}_2 = \mathbf{0}$. Here $C_2\boldsymbol{x}_2 = C_2A_1B_0\boldsymbol{u}_0 \neq \mathbf{0}$.

$$\begin{aligned} &\mathrm{wt}(\boldsymbol{u}_0) + \mathrm{wt}(\boldsymbol{u}_1) + \mathrm{wt}(\boldsymbol{u}_2) + \mathrm{wt}(C_0\boldsymbol{x}_0 + D_0\boldsymbol{u}_0) \\ &+ \mathrm{wt}(C_1\boldsymbol{x}_1 + D_1\boldsymbol{u}_1) + \mathrm{wt}(C_2\boldsymbol{x}_2 + D_2\boldsymbol{u}_2) \\ &= 1 + 0 + 0 + 1 + 1 + 1 = 4. \end{aligned}$$

- $\boldsymbol{u}_0 \neq \mathbf{0}$, $\boldsymbol{u}_1 \neq \mathbf{0}$ and $\boldsymbol{u}_2 \neq \mathbf{0}$. Both terms $C_1\boldsymbol{x}_1 + D_1\boldsymbol{u}_1$ and $C_2\boldsymbol{x}_2 + D_2\boldsymbol{u}_2$ can have weight 0 or 1. However, in this case it is not relevant since the maximum minimum weight is always attained.

$$\begin{aligned} &\mathrm{wt}(\boldsymbol{u}_0) + \mathrm{wt}(\boldsymbol{u}_1) + \mathrm{wt}(\boldsymbol{u}_2) + \mathrm{wt}(C_0\boldsymbol{x}_0 + D_0\boldsymbol{u}_0) \\ &+ \mathrm{wt}(C_1\boldsymbol{x}_1 + D_1\boldsymbol{u}_1) + \mathrm{wt}(C_2\boldsymbol{x}_2 + D_2\boldsymbol{u}_2) \\ &= 1 + 1 + 1 + 1 + \mathrm{wt}(C_1\boldsymbol{x}_1 + D_1\boldsymbol{u}_1) + \mathrm{wt}(C_2\boldsymbol{x}_2 + D_2\boldsymbol{u}_2) \geq 4. \end{aligned}$$

- $\boldsymbol{u}_0 \neq \mathbf{0}$, $\boldsymbol{u}_1 = \mathbf{0}$ and $\boldsymbol{u}_2 \neq \mathbf{0}$. The same occurs with $C_2\boldsymbol{x}_2 + D_2\boldsymbol{u}_2$. In this case, this weight is not necessary to attain the bound.

$$\mathrm{wt}(\boldsymbol{u}_0) + \mathrm{wt}(\boldsymbol{u}_1) + \mathrm{wt}(\boldsymbol{u}_2) + \mathrm{wt}(C_0\boldsymbol{x}_0 + D_0\boldsymbol{u}_0)$$

Universitat d'Alacant
Universidad de Alicante

$$+ \operatorname{wt}(C_1\boldsymbol{x}_1 + D_1\boldsymbol{u}_1) + \operatorname{wt}(C_2\boldsymbol{x}_2 + D_2\boldsymbol{u}_2)$$
$$= 1 + 0 + 1 + 1 + 1 + \operatorname{wt}(C_2\boldsymbol{x}_2 + D_2\boldsymbol{u}_2) \geq 4.$$

- $\boldsymbol{u}_0 \neq \mathbf{0}$, $\boldsymbol{u}_1 \neq \mathbf{0}$ and $\boldsymbol{u}_2 = \mathbf{0}$.

$$\operatorname{wt}(\boldsymbol{u}_0) + \operatorname{wt}(\boldsymbol{u}_1) + \operatorname{wt}(\boldsymbol{u}_2) + \operatorname{wt}(C_0\boldsymbol{x}_0 + D_0\boldsymbol{u}_0)$$
$$+ \operatorname{wt}(C_1\boldsymbol{x}_1 + D_1\boldsymbol{u}_1) + \operatorname{wt}(C_2\boldsymbol{x}_2 + D_2\boldsymbol{u}_2)$$
$$= 1 + 1 + 0 + 1 + \operatorname{wt}(C_1\boldsymbol{x}_1 + D_1\boldsymbol{u}_1) + \operatorname{wt}(C_2\boldsymbol{x}_2 + D_2\boldsymbol{u}_2).$$

  In this situation we need that one of those two weights is nonzero. With a similar reasoning to the one above we can check that to obtain this we need $\frac{c_1}{d_1} \neq \frac{a_1}{b_1}$. Then both terms cannot be 0 at the same time and $d_2^c(\mathcal{C})$ is maximized.

If we choose $\mathbb{F}_p$ to have a large enough number of elements we can always construct an MDP code $\mathcal{C}$ whose matrices satisfy the previous conditions.

Example 5.2: In this toy example we show how one can obtain an MDP time-varying convolutional code from 2 time-invariant convolutional codes, not both optimal.

Let $\mathcal{C}$ be a time-varying code of period $\tau = 2$ over $\mathbb{F}_5$ constituted by the following $(4, 1, 1)$ subcodes:

$$\mathcal{C}_0 = (A, B_0, C_0, D), \quad \mathcal{C}_1 = (A, B_1, C_1, D)$$

where

$$A = \begin{bmatrix} 1 \end{bmatrix}, B_0 = \begin{bmatrix} 3 \end{bmatrix}, B_1 = \begin{bmatrix} 1 \end{bmatrix}, C_0 = \begin{bmatrix} 2 \\ 3 \\ 2 \end{bmatrix}, C_1 = \begin{bmatrix} 2 \\ 1 \\ 1 \end{bmatrix}, D = \begin{bmatrix} 4 \\ 3 \\ 4 \end{bmatrix}.$$

In this case $n = 4$ so we only need to check the first condition we obtained, $\frac{d_{1,j}}{c_{1,j}} \neq \frac{d_{1,i}}{c_{1,i}}$ for all i, j. We have $\frac{4}{2} \neq \frac{3}{1} \neq \frac{4}{1}$, so $\mathcal{C}$ is an MDP code. This condition is equivalent

to check that all the 2×2 not trivially zero minors in

$$\begin{bmatrix} D & \\ C_1 B_0 & D \end{bmatrix} = \begin{bmatrix} 4 & \\ 3 & \\ 4 & \\ 1 & 4 \\ 3 & 3 \\ 3 & 4 \end{bmatrix}$$

are nonzero.

However, $\mathcal{C}_0$ is not an MDP code since $d_1^c(\mathcal{C}_0)$ is not always maximal. The bound says that $d_1^c(\mathcal{C}_0) \leq 7$. If we give inputs $\boldsymbol{u}_0 = 1$ and $\boldsymbol{u}_1 = 1$ we have

$$\mathrm{wt}(\boldsymbol{u}_0) + \mathrm{wt}(\boldsymbol{u}_1) + \mathrm{wt}(C_0\boldsymbol{x}_0 + D\boldsymbol{u}_0) + \mathrm{wt}(C_0\boldsymbol{x}_1 + D\boldsymbol{u}_1) = 1 + 1 + 3 + 1 = 6$$

which is lower than the bound. ■

5.3 Conclusion

In this chapter we introduced a method for constructing periodically time-variant convolutional codes where the degree δ of the system and, therefore, the decoding complexity of the code, do not increase. This fact gives a lower complexity of the arithmetic circuits when implementing the model.

We study the minimality conditions of the new time-varying code and give constructive proofs in order to obtain controllable and observable systems. A lower bound on the free distance is given in terms of the free distance of the equivalent time-invariant code. We give comments about the maximality of the column distances of the code and develop the MDP conditions for time-varying codes constructed using $(n, 1, 1)$-convolutional codes. In Example 5.2 we generate an MDP convolutional code of period $\tau = 2$ where one of the former subcodes is not MDP. In conclusion, it is not necessary that the former subcodes are all MDP in order to obtain a time-varying MDP convolutional code.

As an open question, it remains to analyze the conditions on the former subcodes in order to obtain an MDP code for higher values of the parameters.

Universitat d'Alacant
Universidad de Alicante

Bibliography

[1] K. A. S. Abdel-Ghaffar. Generalized iterative decoding for linear block codes on the binary erasure channel. In *Proceedings of the 2007 IEEE International Symposium on Information Theory (ISIT 2007)*, pages 66–70. IEEE, 2007.

[2] M. Aissen, A. Edrei, I. J. Schoenberg and A. Whitney. On the generating functions of totally positive sequences. In *Proceedings of the National Academy of Sciences of the United States of America*, volume 37, pages 303–307. 1951.

[3] T. Ando. Totally positive matrices. *Linear Algebra and its Applications*, **90**: 165–219 (1987).

[4] K. S. Andrews, D. Divsalar, S. Dolinar, J. Hamkins, C. R. Jones and F. Pollara. The development of turbo and LDP codes for deep-space applications. In *Proceedings of the 2007 IEEE International Symposium on Information Theory (ISIT 2007)*, pages 2142–2156. IEEE, 2007.

[5] A. Antoulas. On recursiveness and related topics in linear systems. *IEEE Transactions on Automatic Control*, **31(12)**: 1121–1135 (1986).

[6] M. Arai, R. Yamamoto, A. Yamaguchi, S. Fukumoto and K. Iwasaki. Analysis of using convolutional codes to recover packet losses over burst erasure channel. In *Proceedings of the 2001 Pacific Rim International Symposium on Dependable Computing*, pages 258–265. 2001.

[7] V. Ayala, K. W. and L. San Martin. Control sets and total positivity. *Semigroup Forum*, **69**: 113–140 (2004).

[8] V. B. Balakirsky and B. D. Kudryashov. List decoding for convolutional codes. *Problems of Information Transmission*, **25(1)**: 16–17 (1989).

[9] V. B. Balakirsky. A necessary and sufficient condition for time-variant

Universitat d'Alacant
Universidad de Alicante

convolutional encoders to be noncatastrophic. In G. D. Cohen, S. Litsyn, A. Lobstein and G. Zémor (editors), *Algebraic Coding*, volume 781 of *Lecture Notes in Computer Science*, pages 1–10. Springer-Verlag, Berlin, 1994.

[10] G. P. Calzolani, M. Chiani, F. Chiaraluce, R. Garello and E. Paolini. Channel coding for future space missions: New requirements and trends. In *Proceedings of the 2007 IEEE International Symposium on Information Theory (ISIT 2007)*, pages 2157–2170. IEEE, 2007.

[11] D. J. Costello, Jr. Free distance bounds for convolutional codes. *IEEE Transactions on Information Theory*, **20(3)**: 356–365 (1974).

[12] T. Craven and G. Csordas. A sufficient condition for strict total positivity of a matrix. (1991).

[13] T. Craven and G. Csordas. Complex zero decreasing sequences. *Methods and Applications of Analysis*, **2**: 420–441 (1995).

[14] E. Curtis, D. V. Ingerman and J. Morrow. Circular planar graphs and resistor networks. *Linear Algebra and its Applications*, **283**: 115–150 (1998).

[15] A. Dholakia. *Introduction to Convolutional Codes with Applications*. Kluwer Academic Publishers, Boston, MA, 1994.

[16] D. Divsalar, S. Dolinar and C. Jones. Protograph ldpc codes over burst erasure channels. *Military communications Conference, MILCOM*, pages 1–7 (2006).

[17] A. Edrei. On the generating functions of totally positive sequences II. *Journal d'Analyse Mathématique*, **2**: 104–109 (1952).

[18] A. Edrei. On the generating function of a double infinite totally positive sequence. *Transactions of the American Mathematical Society*, **74**: 367–383 (1953).

[19] A. Edrei. Proof of a conjecture of Schoenberg on the generating function of a totally positive sequence. *Canadian Journal of Mathematics*, **5**: 86–943 (1953).

[20] P. Elias. Coding for noisy channels. In *IRE International Convention Record, pt. 4*, pages 37–46. 1955.

[21] M. A. Epstein. Algebraic decoding for a binary erasure channel. Technical Report 340, Massachusetts Institute of Technology, 1958.

[22] S. M. Fallat and M. I. Gekhtman. Jordan structures of totally nonnegative matrices. *Canadian Journal of Mathematics*, **57**: 82–98 (2005).

[23] F. Fekri, M. Sartipi, R. M. Mersereau and R. W. Schafer. Convo-

Universitat d'Alacant
Universidad de Alicante

lutional codes using finite-field wavelets: time-varying codes and more. *IEEE Transactions on Signal Processing*, **53(5)**: 1881–1896 (2005).

[24] S. Fomin and A. Zelevinsky. Total positivity: tests and parametrizations. (1999).

[25] E. Fornasini and M. E. Valcher. Multidimensional systems with finite support behaviors: Signal structure, generation, and detection. *SIAM Jorunal on Control and Optimization*, **36(2)**: 760–779 (1998).

[26] G. D. Forney, Jr. Structural analysis of convolutional codes via dual codes. *IEEE Transactions on Information Theory*, **19(4)**: 512–518 (1973).

[27] G. D. Forney, Jr. and M. D. Trott. The dynamics of group codes: state spaces, trellis diagrams, and canonical encoders. *IEEE Transactions on Information Theory*, **39(9)**: 1491–1513 (1993).

[28] P. A. Fuhrmann. Algebraic system theory: an anlayst's point of view. *Journal of the Franklin Institute*, **301**: 521–540 (1976).

[29] F. R. Gantmacher and M. G. Krein. *Oszillationsmatrizen, Oszillationskerne und Kleine Schwingungen Mechanischer Systeme*. Akademie-Verlag, Berlin, 1960.

[30] M. Gasca and J. M. Peña. On the characterization of almost strictly totally positive matrices. *Advances in Computational Mathemtics*, **3**: 239–250 (1995).

[31] H. Gluesing-Luerssen, U. Helmke and J. I. Iglesias Curto. Algebraic decoding for doubly cyclic convolutional codes. *Advances in Mathematics of Communications*, **4(1)**: 83–99 (2010).

[32] H. Gluesing-Luerssen and F.-L. Tsang. A matrix ring description for cyclic convolutional codes. *Advances in Mathematics of Communications*, **2(1)**: 55–81 (2008).

[33] H. Gluesing-Luerssen, J. Rosenthal and R. Smarandache. Strongly MDS convolutional codes. *IEEE Transactions on Information Theory*, **52(2)**: 584–598 (2006).

[34] M. Hazewinkel. Moduli and canonical forms for linear dynamical systems III: The algebraic geometric case. In *Proceedings of the 76 Ames Research Center (NASA) Conference on Geometric Control Theory*, pages 291–336. 1977.

[35] W. C. Huffman and V. Pless. *Fundamentals of Error-Correcting Codes*. Cambridge University Press, Cambridge, 2003.

[36] R. Hutchinson. The existence of strongly MDS convolutional codes. *SIAM*

Universitat d'Alacant
Universidad de Alicante

Journal on Control and Optimization, **47(6)**: 2812–2826 (2008).

[37] R. Hutchinson, R. Smarandache and J. Trumpf. On superregular matrices and MDP convolutional codes. *Linear Algebra and its Applications*, **428(11)**: 2585–2596 (2008).

[38] R. Hutchinson, R. Smarandache and J. Trumpf. Superregular matrices and the construction of convolutional codes having a maximum distance profile.

[39] R. Hutchinson, J. Rosenthal and R. Smarandache. Convolutional codes with maximum distance profile. *Systems & Control Letters*, **54(1)**: 53–63 (2005).

[40] J. I. Iglesias Curto. *A Study on Convolutional Codes. Classification, New Families and Decoding.* PhD Thesis, University of Salamanca, 2008.

[41] R. Johannesson and K. S. Zigangirov. Distances and distance bounds for convolutional codes – an overview. In G. Einarsson, T. Ericson, I. Ingemarsson, R. Johannesson, K. Zigangirov and C.-E. Sundberg (editors), *Topics in Coding Theory*, volume 128 of *Lecture Notes in Control and Information Sciences*, pages 109–136. Springer-Verlag, Berlin, 1989.

[42] R. Johannesson and K. S. Zigangirov. *Fundamentals of Convolutional Coding.* IEEE Press, New York, NY, 1999.

[43] R. E. Kalman. Algebraic structure of linear dynamical systems I. The module of σ. In *Proceedings of the National Academy of Sciences of the United States of America*, volume 54, pages 1503–1508. 1965.

[44] R. E. Kalman, P. L. Flab and M. A. Arbib. *Topics in Mathematical System Theory.* McGraw-Hill, New York, 1969.

[45] S. Karlin. *Total positivity.* Standford University Press, Standford, CA, 1968.

[46] O. M. Katkova and A. M. Vishnyakova. On sufficient conditions for the total positivity and for the multiple positivity of matrices.

[47] R. G. Kermode. Scoped hybrid automatic repeat request with forward error correction. In *Proceedings of the ACM SIGCOMM'98 conference on applications, technologies, architectures and protocols for computer communication.*, pages 278–289. ACM, New York, NY, 1998.

[48] B. Kitchens. Symbolic dynamics and convolutional codes. In B. Marcus and J. Rosenthal (editors), *Codes, Systems, and Graphical Models*, pages 347–360. Springer-Verlag, Berlin, 2000.

[49] P. Koev. Accurate eigenvalues and svds of totally nonnegative matrices. *SIAM*

Universitat d'Alacant
Universidad de Alicante

Journal on Matrix Analysis and Applications, **27(1)**: 1–23 (2005).

[50] B. M. Kurkoski, P. H. Siegel and J. K. Wolf. Analysis of convolutional codes on the erasure channel. In *Proceeding of the IEEE International Symposium on Information Theory*, page 458. IEEE, 2004.

[51] P. J. Lee. There are many good periodically time-varying convolutional codes. *IEEE Transactions on Information Theory*, **35(2)**: 460–463 (1989).

[52] S. Lin and D. J. Costello, Jr. *Error Control Coding: Fundamentals and Applications*. Prentice Hall, Englewood Cliffs, NJ, 1983.

[53] D. Lind and B. Marcus. *Symbolic Dynamics and Coding*. Cambridge University Press, New York, NY, 1995.

[54] M. Luby. LT-code. In *Proceeding of the 43rd Annual IEEE Symposium on the Foundations of Computer Science*, pages 271–280. IEEE, 2002.

[55] M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi and D. A. Spielman. Efficient erasure correcting codes. *IEEE Transactions on Information Theory*, **47(2)**: 569–584 (2001).

[56] F. J. MacWilliams and N. J. A. Sloane. *The Theory of Error-Correcting Codes*. North-Holland, Amsterdam, 6th edition, 1988.

[57] B. Marcus. Symbolic dynamics and connections to coding theory, automata theory and system theory. In *Proceedings of Symposia in Applied Mathematics*, volume 50, pages 90–108. American Mathematical Society, 1995.

[58] J. L. Massey and M. Sain. Codes, automata, and continuous systems: explicit interconnections. *IEEE Transactions on Automatic Control*, **12(6)**: 644–650 (1967).

[59] R. J. McEliece. The algebraic theory of convolutional codes. In V. S. Pless and W. C. Huffman (editors), *Handbook of Coding Theory*, pages 1065–1138. Elsevier, North-Holland, 1998.

[60] M. Mooser. Some periodic convolutional codes better than any fixed code. *IEEE Transactions on Information Theory*, **29(5)**: 750–751 (1983).

[61] C. O'Donoghue and C. Burkley. Catastrophicity test for time-varying convolutional encoders. In M. Walker (editor), *Crytography and Coding*, volume 1746 of *Lecture Notes in Computer Science*, pages 153–162. Springer-Verlag, Berlin, 1999.

[62] N. Ogasahara, M. Kobayashi and S. Hirasawa. The construction of periodically time-variant convolutional codes using binary linear block codes.

Universitat d'Alacant
Universidad de Alicante

Electronics and Communications in Japan, Part 3, **90(9)**: 31–40 (2007).

[63] P. Oswald and M. A. Shokrollahi. Capacity-achieving sequences for the erasure channel. *IEEE Transactions on Information Theory*, **48**: 3019–3028 (2002).

[64] V. Paxson. *Measurements and Analysis of End-to-End Internet Dyu.* PhD Thesis, Computer Science Department, University of California, Berkeley, USA, 1997.

[65] P. Piret. *Convolutional Codes, an Algebraic Approach.* MIT Press, Boston, MA, 1988.

[66] C. D. D. Proietti, I. E. Telatar, T. Richardson and R. Urbanke. Finitelength analysis of low-density parity-check codes on the binary erasure channel. *IEEE Transactions on Information Theory*, **48**: 1570–1579 (2002).

[67] M. S. Ravi and J. Rosenthal. A smooth compactification of the space of transfer functions with fixed mcmillan degree. *Acta Applicandae Mathematicae*, **34**: 329–352 (1994).

[68] S. Roman. *Coding and Information Theory.* Springer, New York, NY, 1992.

[69] S. Roman. *Introduction to Coding and Information Theory.* Springer, New York, NY, 1997.

[70] J. Rosenthal. Connections between linear systems and convolutional codes. In B. Marcus and J. Rosenthal (editors), *Codes, Systems and Graphical Models*, volume 123 of *The IMA Volumes in Mathematics and its Applications*, pages 39–66. Springer-Verlag, New York, 2001.

[71] J. Rosenthal, J. Schumacher and E. V. York. On behaviors and convolutional codes. *IEEE Transactions on Information Theory*, **42(6)**: 1881–1891 (1996).

[72] J. Rosenthal and R. Smarandache. Maximum distance separable convolutional codes. *Applicable Algebra in Engineering, Communication and Computing*, **10**: 15–32 (1999).

[73] J. Rosenthal and E. V. York. BCH convolutional codes. *IEEE Transactions on Information Theory*, **45(6)**: 1833–1844 (1999).

[74] R. M. Roth and G. Seroussi. On generator matrices of MDS codes. *IEEE Transactions on Information Theory*, **31(6)**: 826–830 (1985).

[75] I. J. Schoenberg. *Selected papers*, volume 2 of *Contemporary Mathemtacians.* Birkhäuser, Boston, MA, 1988.

Universitat d'Alacant
Universidad de Alicante

[76] M. A. Shokrollahi. Capacity-achieving sequences. *MA Volumes in Mathematics and its Applications*, **123**: 153–166 (2000).

[77] M. A. Shokrollahi, S. Lasse and M. Luby. Multi-stage code generator and decoder for communication systems. *U.S. Patent application number 20030058958*, (2001).

[78] H. Song and J. R. Cruz. Reduced-complexity decoding of q-ary LDPC codes for magnetic recording. *IEEE Transactions on Magnetics*, **39(2)**: 1081–1087 (2003).

[79] B. Strumfels. Totally positive matrices and cyclic polytopes. *Institute for Mathematics and its Applications*, (1987).

[80] C. Thommesen and J. Justesen. Bounds on distances and error exponents of unit memory codes. *IEEE Transactions on Information Theory*, **29(5)**: 637–649 (1983).

[81] M. E. Valcher and E. Fornasini. On 2d finite support convolutional codes: An algebraic approach. *Multidimensional Systems and Signal Processing*, **5**: 231–243 (1994).

[82] E. Vinberg. Real entire functions with prescribed critical values. *Problems in Group Theory and in Homological Algebra*, pages 127–138 (1989).

[83] A. J. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, **13**: 260–269 (1967).

[84] P. A. Weiner. *Multidimensional Convolutional Codes*. PhD Thesis, Department of Mathematics, University of Notre Dame, Indiana, USA, April 1998.

[85] J. C. Willems. From time series to linear system – Part I. Finite dimensional linear time invariant systems. *Automatica*, **22(5)**: 561–580 (1986).

[86] J. C. Willems. Models for dynamics. In U. Kirchgraber and H. O. Walther (editors), *Dynamics Reported*, volume 2, pages 171–269. John Willey & Sons Ltd. and B.G. Teubner, 1989.

[87] J. C. Willems. Paradigms and puzzles in the theory of dynamical systems. *IEEE Transactions on Automatic Control*, **36(3)**: 259–294 (1991).

[88] M. Yang and W. E. Ryan. Performance of efficiently encodable low-density parity-check codes in noise bursts on the EPR4 channel. *IEEE Transactions on Magnetics*, **40(2)**: 507–512 (2004).

[89] Y. Yasuda, K. Kashiki and Y. Hirata. High-rate punctured convolutional

codes for soft decision Viterbi decoding. *IEEE Transactions on Computers*, **32(3)**: 315–319 (1984).

[90] E. V. York. *Algebraic Description and Construction of Error Correcting Codes: A Linear Systems Point of View*. PhD Thesis, Department of Mathematics, University of Notre Dame, Indiana, USA, May 1997.

[91] X. Y. Zhon and A. E. Kamal. Automatic repeat-request protocols and their queuing analysis. *Computer Communications*, **13(5)**: 298–311 (1990).

[92] K. Zigangirov and H. Osthoff. Analysis of global-list decoding for convolutional codes. *European Transactions on Telecommunications*, **4(2)**: 165–173 (1993).

Universitat d'Alacant
Universidad de Alicante

La defensa de la tesis doctoral realizada por Dª Virtudes Tomás Estevan se ha realizado en las siguientes lenguas: ..... y ....., lo que unido al cumplimiento del resto de requisitos establecidos en la Normativa propia de la UA le otorga la mención de "Doctor Europeo".

Alicante, ... de ..... de .....

El Secretario

El Presidente

Universitat d'Alacant
Universidad de Alicante

Universitat d'Alacant
Universidad de Alicante

Reunido el Tribunal que suscribe en el día de la fecha acordó otorgar, por ..... a la Tesis Doctoral de Dª Virtudes Tomás Estevan la calificación de .....

Alicante, ... de ..... de .....

El Secretario El Presidente

Universitat d'Alacant
Universidad de Alicante

Universidad de Alicante
cedip

La presente Tesis de Dª Virtudes Tomás Estevan ha sido registrada con el nº ..... del registro de entrada correspondiente.

Alicante, ... de ..... de .....

El Encargado del Registro