Integral Distinguishers and a 4-Round Key-Recovery Attack on Kuznyechik Without Initial Key Whitening

Nitish Kumar, Ranit Dutta, and Bimal Mandal

Department of Mathematics, Indian Institute of Technology Jodhpur,
Karwar–342030, India
{p23ma0010, p22ma207, bimalmandal}@iitj.ac.in

Abstract. Kuznyechik is a 128-bit block cipher standardized in GOST R 34.12–2015. In this paper We study Kuznyechik from the viewpoint of integral cryptanalysis, i.e., we track how structured multisets of chosen plaintexts propagate through the round functions. Starting from a first-order structure of 2^8 plaintexts (one byte takes all 256 values while the remaining bytes are fixed), we obtain a 2-round distinguisher: after two rounds, every byte position is balanced, meaning that the XOR-sum over the 256 texts equals zero. Next, in the setting without initial key-whitening, we extend this distinguisher to three rounds by applying one inverse round to the original structure to construct a new input set. Finally, we turn the 3-round balanced property into a 4-round key-recovery attack by partially inverting the last round and filtering last-round key-byte guesses using the balanced test; multiple independent structures remove false candidates.

Keywords: Kuznyechik · Integral cryptanalysis · Key recovery

1 Introduction

The widespread deployment of digital services has made efficient and well-studied encryption a basic requirement in modern systems. Symmetric-key primitives, and block ciphers in particular, remain the main tool for high-throughput confidentiality in practice. Kuznyechik is a 128-bit SP-network block cipher standardized in GOST R 34.12–2015 [1] and in use since 2016. Given its standardization and deployment, evaluating reduced-round security under modern cryptanalytic techniques remains of practical and scientific interest. Cryptanalysis aims to assess the strength of cryptographic algorithms by identifying potential weaknesses that could be exploited by an adversary. Over the years, numerous cryptanalytic techniques have been developed for analyzing symmetric block ciphers, including differential cryptanalysis, linear cryptanalysis, algebraic attacks, and integral cryptanalysis. Algebraic attacks attempt to exploit systems of low-degree polynomial equations describing a cipher. In Kuznyechik, the high algebraic degree of the S-box reduces the feasibility of such representations, thereby enhancing resistance to algebraic cryptanalysis [2]. Differential cryptanalysis studies how pairs of plaintexts with a chosen input difference propagate through a block cipher, aiming to predict output differences with non-negligible probability. This paradigm was established by Biham and Shamir [3] and became a foundational technique for analyzing iterated block ciphers. A key conceptual bridge toward integral attacks is to move from reasoning about pairwise differences to reasoning about higher-order derivatives (and, equivalently, cancellation phenomena over structured sets of many texts). Knudsen [4] formalized truncated and higher-order differential viewpoints, showing that when a cipher's round function behaves like a low-degree Boolean polynomial (over $\mathbb{F}_2$), summing (XORing) evaluations over carefully chosen input subspaces can cancel terms and yield deterministic relations after a small number of rounds. This set-based viewpoint appears concretely in the Square family: one chooses a set of 2^8 plaintexts where one byte takes all 256 values while other bytes are held constant. After a few rounds, certain state bytes become *balanced* (their XOR over the set equals zero), yielding a distinguisher and, with additional steps, a key-recovery strategy. This is the classic integral [5] phenomenon.

Knudsen and Wagner [6] later unified and generalized these ideas under the name integral cryptanalysis: an attack methodology that tracks how multisets propagate, predicting when components become constant, active, or balanced, and giving a systematic framework that subsumes square-like arguments as special cases. Each of these techniques explores different structural and statistical properties of encryption algorithms. In contrast to classical cryptanalysis methods that focus on individual plaintext–ciphertext pairs, integral cryptanalysis examines the behavior of the cipher over structured sets of plaintexts. This approach analyzes how specific properties of multiple plaintexts propagate through the encryption rounds and affect the resulting ciphertexts. Various type of cryptanalysis of Kuznyechik mainly targets reduced-round variants or implementation settings. AlTawy and Youssef [7] presented meet-in-the-middle attacks on reduced-round Kuznyechik (reported up to 5 rounds). On the implementation side, AlTawy, Duman, and Youssef [8] studied fault attacks: in a random-byte fault model targeting rounds 7 and 8, they recover the master key with only a few faults on average, and they also analyze a setting with a secret S-box using ineffective fault analysis. Biryukov, Khovratovich, and Perrin [9] developed multiset-algebraic techniques leading to attacks on reduced Kuznyechik (up to 7 rounds out of

Table 1: Comparison with existing public cryptanalysis of Kuznyechik.

Work	Type	Rounds	Data	Time / Memory
AlTawy–Youssef [7]	Meet-in-the-middle	5	2^{113}	$2^{140.3}$ / $2^{153.3}$
Biryukov–Khovratovich–Perrin [9]	Multiset-algebraic	7	2^{128}	2^{155} / 2^{140}
ElSheikh–Youssef [10]	Integral (bit-based division property, MILP)	3	2^{56}	distinguisher
ElSheikh–Youssef [10]	Integral (bit-based division property, MILP)	4	2^{120}	distinguisher
Stănică–Dutta–Mandal [11]	Statistical distinguisher	9	2^{33}	2^{34} / $(2^8-1)\times 2^8$
Ilkhom et al. [12]	Integral cryptanalysis	3	3×2^{15}	2^{20} / 2^{15}
This work	Integral distinguisher (without pre-whitening)	3	2^8 CP	distinguisher
This work	Key-recovery (without pre-whitening)	4	2^9 CP	$\approx 2^{21}$ invS-box + $2^9 L^{-1}$ / $\approx$ 8KB

9). ElSheikh and Youssef [10] proposed an MILP-based method to model the bit-based division property for ciphers with large diffusion layers, and applied it to integral distinguishers of Kuznyechik. They report improved 3-round and 4-round integral distinguishers requiring 2^{56} and 2^{120} chosen texts, respectively. Stănică, Dutta, and Mandal [11] introduced truncated inner c-differential cryptanalysis, overcoming structural obstacles that previously prevented practical c-differential attacks on block ciphers. Using this technique, they demonstrated the first practical statistical distinguisher against full 9-round Kuznyechik (without pre-whitening), achieving a bias of up to 1.7× with data complexity about 2^{33} chosen plaintext pairs. Boykuziev *et al.* [12] proposed an integral cryptanalysis of the 3-phase Kuznyechik cipher by exploiting balanced byte properties across encryption phases, leading to a key-recovery attack.

Contributions. In this paper we studied 3 round integral distinguisher of Kuznyechik cipher without key prewhitening. In this paper we aim to focus on the areas as follows:

- We revisit the known first-order integral property of Kuznyechik and present a concise 2-round distinguisher based on a 2^8-plaintext structure (one active byte, 15 constant bytes), where after two rounds all 16 state bytes are balanced. [12]
- In the no-prewhitening setting, we extend this distinguisher by one round to obtain a 3-round distinguisher. The extension is achieved by constructing a preimage structure via one-round inversion using L^{-1} and S^{-1}.
- We convert the resulting 3-round balanced property into a 4-round key-recovery attack (still without initial key-whitening). By partially inverting the last round and applying byte-wise balanced tests, we recover the last-round key byte-by-byte; multiple independent structures are used to eliminate false candidates.

Organization. The rest of the paper has been organised as follows: Section 3 recalls the notation and the components of Kuznyechik (round function, S/S^{-1}, L/L^{-1}, and key schedule). Section 4 develops the integral distinguishers, including the one-round inversion technique to reach three rounds and presents the 4-round key-recovery attack and its complexity analysis. Section 5 describes limitations of our work. Section 6 concludes the paper.

2 Preliminaries

This section introduces the notation and the components of Kuznyechik that are used in our integral analysis. Since our distinguishers and key-recovery attack are carried out in a reduced-round setting without initial key-whitening, we also fix the exact cipher model studied in the remainder of the paper.

2.1 Notation

Let $\mathbb{F}_2$ be the field with two elements and let $G = \mathbb{F}_{2^8}$. We represent a 128-bit state as a vector of 16 bytes in G^{16}, often arranged as a 4×4 byte matrix in row-major order.

- P and C denote a plaintext and the corresponding ciphertext.
- X_r denotes the internal state after round r.
- For a multiset $P \subseteq G^{16}$ and a byte position i, we write $\bigoplus_{x\in P} x_i$ for the XOR-sum of the i-th byte over P.

We use the standard integral symbols for a byte position across a 2^8-element structure:

- **A** (All): all 256 values occur exactly once,
- **B** (Balanced): the XOR-sum over the 256 values is 0,
- **C** (Constant): the value is identical in all texts,
- **M** (Mixed): none of the above properties is guaranteed.

2.2 Kuznyechik cipher structure

Kuznyechik is a symmetric block cipher standardized under GOST R 34.12–2015. It encrypts 128-bit data blocks using a 256-bit secret key and is based on a substitution–permutation network (SPN) architecture comprising nine rounds and a final key addition. The cipher operates over the finite field $\mathbb{F}_{2^8}$ defined by the irreducible polynomial

$$p(x) = x^8 + x^7 + x^6 + x + 1.$$

Each 128-bit block is represented as a vector of 16 bytes, with each byte interpreted as an element of $\mathbb{F}_{2^8}$.

The S-box. The Kuznyechik substitution box (S-box) is a bijective mapping $S : \mathbb{F}_{2^8} \to \mathbb{F}_{2^8}$, designed to ensure strong nonlinearity and to provide resistance against linear and differential cryptanalysis. The S-box is defined by a 16×16 lookup table, in which each input byte is interpreted as a pair of hexadecimal digits specifying the corresponding row and column indices. The table entries are represented in hexadecimal notation and arranged in row-major order.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	fc	ee	dd	11	cf	6e	31	16	fb	c4	fa	da	23	c5	04	4d
1	e9	77	f0	db	93	2e	99	ba	17	36	f1	bb	14	cd	5f	c1
2	f9	18	65	5a	e2	5c	ef	21	81	1c	3c	42	8b	01	8e	4f
3	05	84	02	ae	e3	6a	8f	a0	06	0b	ed	98	7f	d4	d3	1f
4	eb	34	2c	51	ea	c8	48	ab	f2	2a	68	a2	fd	3a	ce	cc
5	b5	70	0e	56	08	0c	76	12	bf	72	13	47	9c	b7	5d	87
6	15	a1	96	29	10	7b	9a	c7	f3	91	78	6f	9d	9e	b2	b1
7	32	75	19	3d	ff	35	8a	7e	6d	54	c6	80	c3	bd	0d	57
8	df	f5	24	a9	3e	a8	43	c9	d7	79	d6	f6	7c	22	b9	03
9	e0	0f	ec	de	7a	94	b0	bc	dc	e8	28	50	4e	33	0a	4a
A	a7	97	60	73	1e	00	62	44	1a	b8	38	82	64	9f	26	41
B	ad	45	46	92	27	5e	55	2f	8c	a3	a5	7d	69	d5	95	3b
C	07	58	b3	40	86	ac	1d	f7	30	37	6b	e4	88	d9	e7	89
D	e1	1b	83	49	4c	3f	f8	fe	8d	53	aa	90	ca	d8	85	61
E	20	71	67	a4	2d	2b	09	5b	cb	9b	25	d0	be	e5	6c	52
F	59	a6	74	d2	e6	f4	b4	c0	d1	66	af	c2	39	4b	63	b6

Table 2: Kuznyechik S-box

Inverse S-box. The inverse S-box S^{-1} of Kuznyechik is defined as the inverse permutation of the S-box mapping $S : \mathbb{F}_{2^8} \to \mathbb{F}_{2^8}$, that is, for all $x \in \mathbb{F}_{2^8}$, it holds that $S^{-1}(S(x)) = x$. The inverse S-box is specified by the following 16×16 lookup table, where values are given in hexadecimal notation and arranged in row-major order.

Linear transformation layer. The linear diffusion layer L maps G^{16} to G^{16} and is obtained by iterating a bytewise transformation R sixteen times. Let $a = (a_0, \ldots, a_{15}) \in G^{16}$. Then $R(a)$ is defined by

$$\begin{aligned} R(a)_i &= a_{i+1}, \quad i = 0, 1, \ldots, 14, \\ R(a)_{15} &= \bigoplus_{j=0}^{15} c_j a_j, \end{aligned} \tag{1}$$

where multiplication and addition are in $G = \mathbb{F}_{2^8}$, and

$$c = (\texttt{0x94}, \texttt{0x20}, \texttt{0x85}, \texttt{0x10}, \texttt{0xc2}, \texttt{0xc0}, \texttt{0x01}, \texttt{0xfb}, \texttt{0x01}, \texttt{0xc0}, \texttt{0xc2}, \texttt{0x10}, \texttt{0x85}, \texttt{0x20}, \texttt{0x94}, \texttt{0x01}).$$

The overall linear mapping is

$$L(a) = R^{16}(a).$$

In practice, L and L^{-1} are commonly implemented using precomputed tables. The key expansion procedure of Kuznyechik converts a 256-bit master key into a sequence of ten round keys, denoted by $K^{(0)}, K^{(1)}, \ldots, K^{(9)}$, each of length 128 bits. This procedure is based on a Feistel-type construction that performs 32 successive iterations to compute intermediate key states, from which the round keys are subsequently derived. During the expansion, a set of round-dependent constants C_j, for $j = 1, 2, \ldots, 32$, is employed. Each constant is obtained

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	a5	2d	32	8f	0e	30	38	c0	54	e6	9e	39	55	7e	52	91
1	64	03	57	5a	1c	60	07	18	21	72	a8	d1	29	c6	a4	3f
2	e0	27	8d	0c	82	ea	ae	b4	9a	63	49	e5	42	e4	15	b7
3	c8	06	70	9d	41	75	19	c9	aa	fc	4d	bf	2a	73	84	d5
4	c3	af	2b	86	a7	b1	b2	5b	46	d3	9f	fd	d4	0f	9c	2f
5	9b	43	ef	d9	79	b6	53	7f	c1	f0	23	e7	25	5e	b5	1e
6	a2	df	a6	fe	ac	22	f9	e2	4a	bc	35	ca	ee	78	05	6b
7	51	e1	59	a3	f2	71	56	11	6a	89	94	65	8c	bb	77	3c
8	7b	28	ab	d2	31	de	c4	5f	cc	cf	76	2c	b8	d8	2e	36
9	db	69	b3	14	95	be	62	a1	3b	16	66	e9	5c	6c	6d	ad
A	37	61	4b	b9	e3	ba	f1	a0	85	83	da	47	c5	b0	33	fa
B	96	6f	6e	c2	f6	50	ff	5d	a9	8e	17	1b	97	7d	ec	58
C	f7	1f	fb	7c	09	0d	7a	67	45	87	dc	e8	4f	1d	4e	04
D	eb	f8	f3	3e	3d	bd	8a	88	dd	cd	0b	13	98	02	93	80
E	90	d0	24	34	cb	ed	f4	ce	99	10	44	40	92	3a	01	26
F	12	1a	48	68	f5	81	8b	c7	d6	20	0a	08	00	4c	d7	74

Table 3: Kuznyechik inverse S-box

by applying the linear transformation L to a 16-byte basis vector in which only the j-th byte is set to one, while all remaining bytes are zero. The complete collection of these constants, represented in hexadecimal notation and arranged in row-major order for each 16-byte vector, is provided below.

$$
\begin{aligned}
C_1 &= (\texttt{6e}, \texttt{a2}, 69, \texttt{2a}, \texttt{e9}, 19, 37, \texttt{7e}, \texttt{dc}, \texttt{fe}, 15, 41, 28, \texttt{d9}, 21, 66),\\
C_2 &= (\texttt{b7}, \texttt{5f}, 82, 79, 91, 13, \texttt{dd}, \texttt{4c}, \texttt{b5}, \texttt{8f}, \texttt{9f}, 85, \texttt{1d}, \texttt{ad}, 20, 69),\\
C_3 &= (90, 94, \texttt{4a}, 96, 96, 48, \texttt{6c}, \texttt{a6}, 64, \texttt{2f}, \texttt{ce}, \texttt{3b}, 25, 21, 85, 86),\\
C_4 &= (96, 68, \texttt{8a}, 92, 13, 11, \texttt{bf}, 55, 02, 16, 48, \texttt{0a}, \texttt{2c}, 78, 68, 15),\\
C_5 &= (\texttt{3a}, \texttt{6c}, 54, 74, \texttt{4b}, 39, 52, \texttt{7c}, \texttt{fb}, 98, 45, \texttt{9c}, \texttt{ed}, 38, \texttt{e4}, \texttt{9e}),\\
C_6 &= (39, 25, 23, \texttt{4a}, 94, 11, 84, 57, \texttt{8e}, 24, 69, 93, \texttt{0f}, \texttt{8b}, 73, 93),\\
C_7 &= (\texttt{bc}, \texttt{4f}, 65, 76, \texttt{6e}, \texttt{5f}, 17, \texttt{dd}, 03, \texttt{6f}, 38, 07, 57, 24, 40, 90),\\
C_8 &= (97, \texttt{5c}, \texttt{6f}, 68, 11, \texttt{dd}, 66, \texttt{0b}, 59, \texttt{4a}, 27, 15, 81, \texttt{3d}, 76, \texttt{1c}),\\
C_9 &= (\texttt{2a}, \texttt{2f}, 56, \texttt{7b}, 82, 48, \texttt{2f}, 99, 83, \texttt{5a}, 49, 68, 12, 23, \texttt{2f}, \texttt{2a}),\\
C_{10} &= (\texttt{a4}, \texttt{6b}, \texttt{5d}, \texttt{a6}, 85, 33, \texttt{7f}, \texttt{5e}, \texttt{4b}, 77, 47, 32, \texttt{a7}, 17, 86, 42),\\
C_{11} &= (85, \texttt{2c}, 90, \texttt{2e}, \texttt{a7}, \texttt{3d}, \texttt{4f}, 55, 91, 46, 77, 53, 56, 05, 04, 32),\\
C_{12} &= (\texttt{7e}, \texttt{b3}, 92, 91, 33, 05, 27, 36, 35, 27, \texttt{9e}, \texttt{8b}, \texttt{5b}, \texttt{6e}, 47, 84),\\
C_{13} &= (90, \texttt{ca}, \texttt{4d}, 21, 85, 05, \texttt{d8}, 47, 98, 85, 06, \texttt{bb}, \texttt{6b}, \texttt{d0}, 17, \texttt{8f}),\\
C_{14} &= (94, 44, \texttt{3a}, 81, 25, \texttt{0e}, 96, \texttt{0b}, 62, 56, 24, 68, 84, 01, \texttt{0d}, 06),\\
C_{15} &= (\texttt{a1}, 05, 66, 14, 78, \texttt{bd}, 52, \texttt{db}, 42, 33, \texttt{cd}, \texttt{9e}, \texttt{2d}, 15, \texttt{8a}, \texttt{f1}),\\
C_{16} &= (40, 21, 05, \texttt{5b}, 73, \texttt{8c}, \texttt{5f}, \texttt{5f}, 57, 09, 36, 98, 31, 31, 73, 05)
\end{aligned}
$$

$$C_{17} = (\texttt{2f}, \texttt{33}, \texttt{87}, \texttt{86}, \texttt{67}, \texttt{03}, \texttt{05}, \texttt{32}, \texttt{92}, \texttt{f8}, \texttt{bc}, \texttt{41}, \texttt{13}, \texttt{06}, \texttt{95}, \texttt{26}),$$
$$C_{18} = (\texttt{6c}, \texttt{04}, \texttt{9f}, \texttt{f0}, \texttt{9b}, \texttt{54}, \texttt{8a}, \texttt{18}, \texttt{06}, \texttt{05}, \texttt{66}, \texttt{85}, \texttt{47}, \texttt{65}, \texttt{68}, \texttt{68}),$$
$$C_{19} = (\texttt{51}, \texttt{9c}, \texttt{24}, \texttt{49}, \texttt{28}, \texttt{69}, \texttt{96}, \texttt{26}, \texttt{87}, \texttt{35}, \texttt{89}, \texttt{67}, \texttt{04}, \texttt{91}, \texttt{99}, \texttt{a0}),$$
$$C_{20} = (\texttt{92}, \texttt{82}, \texttt{11}, \texttt{87}, \texttt{1a}, \texttt{35}, \texttt{9e}, \texttt{d6}, \texttt{28}, \texttt{68}, \texttt{21}, \texttt{9b}, \texttt{92}, \texttt{05}, \texttt{67}, \texttt{8b}),$$
$$C_{21} = (\texttt{81}, \texttt{b3}, \texttt{5c}, \texttt{12}, \texttt{74}, \texttt{26}, \texttt{2e}, \texttt{93}, \texttt{6a}, \texttt{66}, \texttt{7f}, \texttt{dc}, \texttt{94}, \texttt{58}, \texttt{7d}, \texttt{13}),$$
$$C_{22} = (\texttt{1a}, \texttt{35}, \texttt{9e}, \texttt{d6}, \texttt{28}, \texttt{68}, \texttt{21}, \texttt{9b}, \texttt{92}, \texttt{05}, \texttt{67}, \texttt{8b}, \texttt{81}, \texttt{b3}, \texttt{5c}, \texttt{12}),$$
$$C_{23} = (\texttt{74}, \texttt{26}, \texttt{2e}, \texttt{93}, \texttt{6a}, \texttt{66}, \texttt{7f}, \texttt{dc}, \texttt{94}, \texttt{58}, \texttt{7d}, \texttt{13}, \texttt{75}, \texttt{61}, \texttt{ba}, \texttt{13}),$$
$$C_{24} = (\texttt{99}, \texttt{34}, \texttt{96}, \texttt{88}, \texttt{23}, \texttt{9b}, \texttt{56}, \texttt{16}, \texttt{80}, \texttt{91}, \texttt{4e}, \texttt{fc}, \texttt{22}, \texttt{2d}, \texttt{3a}, \texttt{92}),$$
$$C_{25} = (\texttt{69}, \texttt{85}, \texttt{28}, \texttt{83}, \texttt{98}, \texttt{47}, \texttt{28}, \texttt{69}, \texttt{96}, \texttt{26}, \texttt{87}, \texttt{35}, \texttt{89}, \texttt{67}, \texttt{04}, \texttt{91}),$$
$$C_{26} = (\texttt{99}, \texttt{a0}, \texttt{92}, \texttt{82}, \texttt{11}, \texttt{87}, \texttt{1a}, \texttt{35}, \texttt{9e}, \texttt{d6}, \texttt{28}, \texttt{68}, \texttt{21}, \texttt{9b}, \texttt{92}, \texttt{05}),$$
$$C_{27} = (\texttt{67}, \texttt{8b}, \texttt{81}, \texttt{b3}, \texttt{5c}, \texttt{12}, \texttt{74}, \texttt{26}, \texttt{2e}, \texttt{93}, \texttt{6a}, \texttt{66}, \texttt{7f}, \texttt{dc}, \texttt{94}, \texttt{58}),$$
$$C_{28} = (\texttt{7d}, \texttt{13}, \texttt{75}, \texttt{61}, \texttt{ba}, \texttt{13}, \texttt{74}, \texttt{26}, \texttt{2e}, \texttt{93}, \texttt{6a}, \texttt{66}, \texttt{7f}, \texttt{dc}, \texttt{94}, \texttt{58}),$$
$$C_{29} = (\texttt{7d}, \texttt{13}, \texttt{99}, \texttt{34}, \texttt{96}, \texttt{88}, \texttt{23}, \texttt{9b}, \texttt{56}, \texttt{16}, \texttt{80}, \texttt{91}, \texttt{4e}, \texttt{fc}, \texttt{22}, \texttt{2d}),$$
$$C_{30} = (\texttt{3a}, \texttt{92}, \texttt{69}, \texttt{85}, \texttt{28}, \texttt{83}, \texttt{98}, \texttt{47}, \texttt{28}, \texttt{69}, \texttt{96}, \texttt{26}, \texttt{87}, \texttt{35}, \texttt{89}, \texttt{67}),$$
$$C_{31} = (\texttt{04}, \texttt{91}, \texttt{99}, \texttt{a0}, \texttt{92}, \texttt{82}, \texttt{11}, \texttt{87}, \texttt{1a}, \texttt{35}, \texttt{9e}, \texttt{d6}, \texttt{28}, \texttt{68}, \texttt{21}, \texttt{9b}),$$
$$C_{32} = (\texttt{92}, \texttt{05}, \texttt{67}, \texttt{8b}, \texttt{81}, \texttt{b3}, \texttt{5c}, \texttt{12}, \texttt{74}, \texttt{26}, \texttt{2e}, \texttt{93}, \texttt{6a}, \texttt{66}, \texttt{7f}, \texttt{dc}).$$

Key schedule algorithm

Algorithm 1 Kuznyechik key schedule

Require: 256-bit master key $K = K_1 \parallel K_0$
Ensure: Round keys $K^{(0)}, \ldots, K^{(9)}$
1: $K^{(0)} \leftarrow K_1$
2: $K^{(1)} \leftarrow K_0$
3: $(a_1, a_0) \leftarrow (K_1, K_0)$
4: **for** $j = 1$ to 32 **do**
5: $(a_1, a_0) \leftarrow F(a_1, a_0, C_j)$
F: S-box $\rightarrow L \rightarrow \oplus C_j$
6: **if** $j \bmod 8 = 0$ **then**
7: $t \leftarrow j/8$
8: $K^{(2t)} \leftarrow a_1$
9: $K^{(2t+1)} \leftarrow a_0$
10: **end if**
11: **end for**

2.3 Cipher model and attacked variant

State space and basic operations. Kuznyechik operates on 128-bit blocks represented as vectors of 16 bytes over $G = \mathbb{F}_{2^8}$, i.e., $X \in G^{16}$ (often arranged as a 4×4 matrix in row-major order). The nonlinear layer S applies the 8-bit S-box independently to each byte, and L denotes the linear diffusion layer defined in Section 2.

Standard cipher [1]. The standard Kuznyechik specification applies an initial key addition (pre-whitening), then 9 rounds, and finally a last key addition. Since our analysis focuses on integral properties of the round function, we only use the standard definition as reference and explicitly define below the reduced-round variant considered in this work.

Reduced-round variant studied in this paper. Throughout the remainder of this paper we study a reduced-round setting in which the initial pre-whitening step is omitted. Concretely, instead of $X_0 = P \oplus K_0$ we set

$$X_0 = P, \tag{2}$$

and for rounds $r \geq 1$ we use the round update

$$X_r = L\big(S(X_{r-1})\big) \oplus K_r. \tag{3}$$

That is, each round applies the S-box layer, then the linear diffusion layer, and finally adds the round key. This round order matches our distinguisher construction and key-recovery derivation. Note that for any fixed round key K_r, the map $X \mapsto X \oplus K_r$ adds the same constant to every element of a structure and therefore does not change the type-level integral classification (**A**, **B**, **C**, **M**).

Reduced-round encryption and decryption. Algorithm 2 summarizes the r-round encryption of the variant (2)–(3), and Algorithm 3 gives the corresponding decryption.

Algorithm 2 Reduced-round encryption used in this paper (without pre-whitening)

Require: Plaintext $P \in G^{16}$ and round keys $K_1, \ldots, K_r \in G^{16}$
Ensure: Ciphertext $C \in G^{16}$
$X_0 \leftarrow P$ ▷ pre-whitening $P \oplus K_0$ is omitted
for $j = 1$ to r **do**
  $X_j \leftarrow L(S(X_{j-1}))$
  $X_j \leftarrow X_j \oplus K_j$
end for
return $C \leftarrow X_r$

Algorithm 3 Reduced-round decryption used in this paper (without pre-whitening)

Require: Ciphertext $C \in G^{16}$ and round keys $K_1, \ldots, K_r \in G^{16}$
Ensure: Plaintext $P \in G^{16}$
$X_r \leftarrow C$
for $j = r$ down to 1 **do**
  $X_{j-1} \leftarrow X_j \oplus K_j$
  $X_{j-1} \leftarrow S^{-1}(L^{-1}(X_{j-1}))$
end for
return $P \leftarrow X_0$

3 Integral distinguisher

Let $G = \mathbb{F}_{2^8}$ and represent the internal state as a vector of 16 bytes in G^{16} (arranged as a 4×4 matrix in row-major order). For a multiset $P \subseteq G^{16}$ and byte position $i \in \{0, 1, \ldots, 15\}$, define the XOR-sum

$$\Sigma_i(P) \;=\; \bigoplus_{x \in P} x_i.$$

We use the standard integral symbols for a byte position over a 2^8-element structure: **A** (all values), **B** (balanced, XOR-sum $= 0$), **C** (constant), and **M** (mixed/unknown).

Cipher model used for distinguishers. In Sections 3–4 we use the reduced-round variant without initial key-whitening for the case of obtaining 3 round distinguisher:

$$X_r = L(S(X_{r-1})) \oplus K_r, \qquad r \geq 1.$$

XOR with a fixed constant does not affect the **A**/**B**/**C**/**M** classification and does not change balancedness, hence we ignore the final "$\oplus K_r$" step when tracking types. For 2 round distinguisher we used the usual cipher model i.e. $X_r = L(S(X_{r-1} \oplus K_0)) \oplus K_r, \qquad r \geq 1.$

Now, we are going to present some results related to obtaining balanced property of integral distinguisher and key independent part, we will discuss in next subsection.

Lemma 1 (XOR over the byte space). *For $G = \mathbb{F}_{2^8}$, $\bigoplus_{a \in G} a = 0$. Moreover, for any permutation $f : G \to G$,*

$$\bigoplus_{a \in G} f(a) = 0.$$

Lemma 2 (Linearity preserves balancedness). *Let $G = \mathbb{F}_{2^8}$ and let $L : G^{16} \to G^{16}$ be $\mathbb{F}_2$-linear. For a multiset $P \subseteq G^{16}$, define its image multiset $L(P) = \{\, L(x) \mid x \in P \,\}$ (with multiplicities), and for each byte position j define $\Sigma_j(P) = \bigoplus_{x \in P} x_j$. Denote the i-th output byte of L as*

$$(L(x))_i = \bigoplus_{\ell=0}^{15} m_{i,\ell}\, x_\ell, \qquad m_{i,\ell} \in G.$$

Then for every $i \in \{0, 1, \ldots, 15\}$,

$$\Sigma_i(L(P)) = \bigoplus_{\ell=0}^{15} m_{i,\ell}\, \Sigma_\ell(P).$$

In particular, if $\Sigma_\ell(P) = 0$ *for all* ℓ*, then* $\Sigma_i(L(P)) = 0$ *for all* i*.*

Proof. By definition,

$$\Sigma_i(L(P)) = \bigoplus_{x \in P} (L(x))_i = \bigoplus_{x \in P} \left(\bigoplus_{\ell=0}^{15} m_{i,\ell} x_\ell \right).$$

Reordering XOR terms gives

$$\Sigma_i(L(P)) = \bigoplus_{\ell=0}^{15} \left(\bigoplus_{x \in P} m_{i,\ell} x_\ell \right).$$

Since multiplication in G distributes over addition, $\bigoplus_{x \in P} m_{i,\ell} x_\ell = m_{i,\ell} \left(\bigoplus_{x \in P} x_\ell \right) = m_{i,\ell} \Sigma_\ell(P)$. Substituting yields the claimed identity, and the "in particular" statement follows immediately.

3.1 Chosen-plaintext structure

We consider a first-order integral consisting of 2^8 chosen plaintexts where only one byte varies and the remaining bytes are fixed constants. Concretely, we take

$$\mathcal{P} = \left\{ P^{(t)} \in G^{16} \;\middle|\; P_0^{(t)} = t,\ t \in G,\ \text{and } P_j^{(t)} = c_j\ \forall j \neq 0 \right\},$$

which corresponds to the type pattern

$$\begin{pmatrix} A & C & C & C \\ C & C & C & C \\ C & C & C & C \\ C & C & C & C \end{pmatrix}.$$

3.2 Two-round integral distinguisher

Let $\mathcal{P}$ be the structure from Section 3.1. After one round, the S-box layer S preserves the **A** and **C** properties because S is a byte permutation (actually same phenomena holds for key addition). The subsequent linear layer L diffuses the activity to all bytes (this full diffusion can be verified by a single evaluation of $L(e_0)$, where e_0 is the unit vector with a 1 in byte position 0 and zeros elsewhere). Hence after one round the state is of type A^{16} as again key addition does not make any effect.

At the beginning of round 2, each byte position takes all 256 values. Applying the S-box keeps this property, it remains same. Lemma 2 implies that L preserves byte wise balancedness. Actually this the first time bytes become balanced but not satisfying ALL property. Again as key addition invariance property, after two rounds all 16 bytes are balanced, yielding the 2-round key independent distinguisher

$$(A, C^{15}) \xrightarrow{\text{1 round}} A^{16} \xrightarrow{\text{2 rounds}} B^{16}.$$

Detailed structure of distinguisher for 2 round is given in Appendix B. Kuznyechik reduced to two rounds can be distinguished from a random permutation using the first-order structure $\mathcal{P}$ of size 2^8. The distinguisher has data complexity 2^8 and time complexity 2^8 reduced-round encryptions. For a random permutation, the probability that all 16 output bytes are balanced is negligible.

3.3 From the 2-round distinguisher to the 3-round distinguisher via one-round inversion

We extend the 2-round integral distinguisher to three rounds in the setting without initial key-whitening, i.e., we omit the pre-whitening step $X_0 = P \oplus K_0$ and set $X_0 = P$. In this setting, the reduced-round encryption is

$$X_r = L\big(S(X_{r-1})\big) \oplus K_r, \qquad r \geq 1, \tag{4}$$

where each round applies the S-box layer, the linear layer, and then adds the round key. Since $\oplus K_r$ adds the same constant to every element of the structure, it preserves the type-level integral properties (**A**, **B**, **C**, **M**) tracked in this work.

$$\begin{pmatrix} A & A & A & A \\ A & A & A & A \\ A & A & A & A \\ A & A & A & A \end{pmatrix} \xleftarrow{S^{-1}} \begin{pmatrix} A & A & A & A \\ A & A & A & A \\ A & A & A & A \\ A & A & A & A \end{pmatrix} \xleftarrow{L^{-1}} \begin{pmatrix} A & C & C & C \\ C & C & C & C \\ C & C & C & C \\ C & C & C & C \end{pmatrix}$$

Fig. 1: Illustration of one-round inversion used to construct the 3-round preimage structure.

Preimage construction. Let X be a multiset of 2^8 internal states that we want to realize after the first nonlinear and linear layers, i.e., right before the addition of K_1. We construct the corresponding plaintext multiset as

$$\mathcal{P}' \;=\; \{\, P' = S^{-1}(L^{-1}(x)) \mid x \in X \,\}. \tag{5}$$

Then for every $P' \in \mathcal{P}'$ we have $L(S(P')) = x$, hence one forward round maps $\mathcal{P}'$ to the intended internal multiset X at the K_1-addition boundary.

Three-round distinguisher. Let $\mathcal{P}$ denote the original 2^8-element first-order structure used for the 2-round distinguisher (one byte takes all values, others are fixed). By Section 3, encrypting $\mathcal{P}$ for two rounds yields a balanced state in every byte position. To obtain a 3-round distinguisher, we choose X in (5) to be the internal multiset corresponding to $\mathcal{P}$ at the input of the $\oplus K_1$ step (equivalently, after the first $L \circ S$). By construction, one forward round maps $\mathcal{P}'$ to $\mathcal{P}$ at this round boundary, and thus the 2-round balanced property applies to the remaining two rounds. Consequently, the state after three rounds is balanced in all 16 byte positions.

Type-level backward propagation (illustration). For intuition, Figure 1 illustrates the effect of one-round inversion on the type pattern (ignoring the key addition). A complete step-by-step trace through $R_{16}^{-1}, \ldots, R_1^{-1}$ is provided in Appendix A.

Experimental verification. We implemented (5) and the reduced-round encryption (4) in Python/Sage to verify the predicted behavior. After each round, each byte position is classified as `A` (all values), `C` (constant), `B` (balanced XOR-sum), or `M` (mixed), and printed as a 4×4 grid. The traces confirm that the constructed preimage structure yields a 3-round bytewise-balanced state as predicted.

4 Key-recovery attack on 4 rounds (without initial key-whitening)

In this section we turn the 3-round integral distinguisher into a key-recovery attack on 4 rounds of Kuznyechik in the setting without initial key-whitening. Concretely, we omit the pre-whitening step $X_0 = P \oplus K_0$ of the standard cipher and set $X_0 = P$. The reduced 4-round encryption is therefore

$$X_r = L\big(S(X_{r-1})\big) \oplus K_r, \qquad r = 1, 2, 3, 4, \tag{6}$$

where each round applies the S-box layer, the linear layer, and then adds the round key. Since $\oplus K_r$ adds the same constant to every element of a structure, it preserves the integral-type properties used in our distinguisher.

4.1 Attack methodology

Let $\mathcal{P}$ be an input multiset of size 2^8 constructed as in Section 3, and let $\mathcal{C} = \{C^{(t)}\}_{t=1}^{2^8}$ be the corresponding ciphertext multiset after 4 rounds. By the 3-round distinguisher, the state after 3 rounds is bytewise balanced; we denote it by $X_3^{(t)}$. The last round of (6) can be written as

$$C^{(t)} = L\big(S(X_3^{(t)})\big) \oplus K_4.$$

Applying L^{-1} to both sides yields

$$L^{-1}(C^{(t)}) = S(X_3^{(t)}) \oplus L^{-1}(K_4).$$

Define the transformed last-round key

$$K_4' \;=\; L^{-1}(K_4), \qquad \text{and set} \qquad Y^{(t)} = L^{-1}(C^{(t)}).$$

Then for each byte position $i \in \{0, 1, \ldots, 15\}$ we obtain

$$S^{-1}\big(Y_i^{(t)} \oplus K_4'[i]\big) = X_{3,i}^{(t)}.$$

Since the multiset $\{X_3^{(t)}\}$ is balanced in every byte position, we get the key-dependent balanced equation

$$\bigoplus_{t=1}^{2^8} S^{-1}\big(Y_i^{(t)} \oplus K_4'[i]\big) = 0 \qquad \text{for all } i \in \{0, 1, \ldots, 15\}. \tag{7}$$

This allows us to recover $K_4'[i]$ independently for each byte position by testing all 2^8 candidates.

4.2 Byte-wise recovery procedure

For a fixed byte position i, and for each guess $g \in \mathbb{F}_{2^8}$, define

$$T_i(g) \;=\; \bigoplus_{t=1}^{2^8} S^{-1}\big(Y_i^{(t)} \oplus g\big).$$

By (7), the correct key byte satisfies $T_i(K_4'[i]) = 0$. We keep all guesses g such that $T_i(g) = 0$ as candidates for $K_4'[i]$.

To eliminate false candidates, we repeat the same procedure with a second independent structure $\mathcal{P}'$ of size 2^8, producing a second ciphertext multiset $\mathcal{C}'$ and corresponding values $Y'^{(t)} = L^{-1}(C'^{(t)})$. We intersect the candidate sets obtained from the two structures for each byte.

Finally, after recovering all 16 bytes of K_4', we compute the actual last-round key as

$$K_4 = L(K_4').$$

4.3 Algorithm description

In this subsection, we present the main algorithm used in our practical key-recovery attack. It describes how the balanced property of the 3-round distinguisher is exploited to recover the last-round key bytes in the 4-round setting.

Algorithm 4 Recovery of K_4' using two 2^8-structures (without pre-whitening)

Require: Two ciphertext multisets $\mathcal{C}_1 = \{C_1^{(t)}\}_{t=1}^{2^8}$ and $\mathcal{C}_2 = \{C_2^{(t)}\}_{t=1}^{2^8}$ obtained by encrypting two independent 2^8-element structures for 4 rounds
Ensure: The transformed last-round key $K_4' = L^{-1}(K_4)$

Compute $Y_1^{(t)} \leftarrow L^{-1}(C_1^{(t)})$ for all $t = 1, 2, \ldots, 2^8$
Compute $Y_2^{(t)} \leftarrow L^{-1}(C_2^{(t)})$ for all $t = 1, 2, \ldots, 2^8$
for $i = 0$ to 15 **do**
    $\mathcal{G}_i \leftarrow \emptyset$
    for $g = 0$ to 255 **do**
        $s \leftarrow 0$
        for $t = 1$ to 2^8 **do**
            $s \leftarrow s \oplus S^{-1}\big(Y_{1,i}^{(t)} \oplus g\big)$
        end for
        if $s = 0$ **then**
            $\mathcal{G}_i \leftarrow \mathcal{G}_i \cup \{g\}$
        end if
    end for
    $\mathcal{G}_i' \leftarrow \emptyset$
    for $g \in \mathcal{G}_i$ **do**
        $s \leftarrow 0$
        for $t = 1$ to 2^8 **do**
            $s \leftarrow s \oplus S^{-1}\big(Y_{2,i}^{(t)} \oplus g\big)$
        end for
        if $s = 0$ **then**
            $\mathcal{G}_i' \leftarrow \mathcal{G}_i' \cup \{g\}$
        end if
    end for
    Set $K_4'[i]$ to the unique element of $\mathcal{G}_i'$; ▷ if needed, use additional structures to break ties
end for
return K_4'

4.4 Experimental validation

We implemented the reduced-round variant (6) and the key test (7) in Sage/Python over $\mathbb{F}_{2^8}$ with modulus $p(x) = x^8 + x^7 + x^6 + x + 1$. Round keys $K_1, \ldots, K_4$ were generated using the standard Kuznyechik key schedule, while the initial pre-whitening was omitted (i.e., $X_0 = P$). For two independent 2^8-structures constructed via the preimage method (5), the attack uniquely recovers all 16 bytes of $K'_4 = L^{-1}(K_4)$; applying L yields the round key K_4. The implementation also confirms that one structure may leave multiple byte candidates, while intersecting two structures gives a unique candidate per byte.

Example. For the master key `0x8899aabbccddeeff0011223344556677fedcba98765432100123456789abcdef`, we obtain $K'_4 = (\texttt{e6}, \texttt{0f}, \texttt{ce}, \texttt{12}, \texttt{ef}, \texttt{b4}, 46, 59, 85, 87, \texttt{5c}, \texttt{be}, \texttt{a0}, 20, \texttt{1d}, \texttt{6e})$, which matches the direct computation of $L^{-1}(K_4)$.

4.5 Complexity and success probability

The data complexity is two first-order structures, i.e., $2 \cdot 2^8 = 2^9$ chosen plaintexts and the same number of 4-round encryptions. The offline phase first computes $Y^{(t)} = L^{-1}(C^{(t)})$ for all ciphertexts, which costs 2^9 evaluations of L^{-1}. The dominant step is the balanced test: per structure, for each of the 16 byte positions we test 2^8 guesses, and each test processes 2^8 texts, resulting in $16 \cdot 2^8 \cdot 2^8 = 2^{20}$ bytewise inverse S-box evaluations. Using two structures, the dominant offline cost is therefore 2^{21} bytewise inverse S-box evaluations, plus 2^9 evaluations of L^{-1}. The memory requirement is about $2 \cdot 2^8 \cdot 16 \approx 8$ KB to store the Y-states.

For the correct key byte the balanced equation holds deterministically. For an incorrect guess, the balanced test holds with probability about 2^{-8} per structure, so the expected number of surviving candidates per byte is $2^8 \cdot 2^{-8} = 1$ after one structure and $2^8 \cdot 2^{-16} = 2^{-8}$ after two structures, yielding unique recovery with high probability.

5 Discussion and limitations

Role of the without pre-whitening setting. Our 3-round distinguisher is obtained by extending the known 2-round first-order integral distinguisher with one additional backward round (one-round inversion). This construction critically uses the reduced-round variant without initial key-whitening. In the standard Kuznyechik specification, the unknown prewhitening key addition $P \oplus K_0$ prevents us from generating the required preimage structure, hence the backward extension is not directly applicable. We experimentally verified that with the same 2^8-plaintext structure we do not obtain a 3-round integral distinguisher in the presence of prewhitening with key independency.

Data requirements for higher-round distinguishers. Our analysis is restricted to first-order structures of size 2^8 and their backward-extended variants. To obtain distinguishers for substantially more rounds, one typically needs much larger plaintext sets and higher-order properties. In particular, degree-based approaches (e.g., using algebraic degree bounds) imply that integral properties may require structures of size as large as 2^{116} (or larger) to appear at deeper rounds, as explicitly analyzed in the multiset-algebraic/degree-based line of work by Biryukov et al. [9]. This is also reflected by division-property based searches: ElSheikh and Youssef [10] report 4-round integral distinguishers for Kuznyechik at data around 2^{120} and for 3 round it is 2^{56}. Consequently, extending our distinguisher beyond three rounds in the standard model is expected to require prohibitive data complexity.

Limits of extending key recovery beyond 4 rounds. We also attempted to extend our key-recovery attack beyond 4 rounds by adding one more round on the ciphertext side. For 5 rounds (still without prewhitening), after peeling the last round via L^{-1} we obtain $Y = L^{-1}(C) = S(X_4) \oplus K'_5$ with $K'_5 = L^{-1}(K_5)$, hence $X_4 = S^{-1}(Y \oplus K'_5)$. The next step required to reach the balanced boundary involves applying L^{-1} to X_4, but Kuznyechik's diffusion is global: each byte of $L^{-1}(X_4)$ depends on all 16 bytes of X_4, and therefore on all 16 bytes of K'_5. As a result, even a single-byte balanced test at the X_3 boundary is not separable by key bytes and inherently involves the full 2^{128} space of K'_5 (and additionally the relevant byte(s) of K'_4 if not already recovered). Thus, the time complexity for extending the attack to 5 rounds is at least $\Omega(2^{128})$ basic operations, making it infeasible. We therefore stop at 4 rounds.

6 Conclusion

We studied Kuznyechik under integral cryptanalysis in a reduced-round model without initial key-whitening. Starting from a first-order 2^8 structure (one active byte), the round function exhibits the expected 2-round

bytewise-balanced property. Exploiting the absence of pre-whitening, we constructed a suitable preimage multiset via one-round inversion, which yields a 3-round bytewise-balanced distinguisher in this setting.

We then converted the 3-round balanced property into a 4-round key-recovery attack by partially inverting the last round and testing transformed last-round key bytes independently using a balanced XOR condition. The resulting attack has low data complexity (two 2^8 structures) and a small offline cost dominated by bytewise inverse S-box evaluations.

Finally, we explained why extending the attack beyond 4 rounds becomes infeasible with the same approach: due to Kuznyechik's full diffusion, peeling one more round couples the balanced test to (essentially) the entire last-round subkey, leading to prohibitive complexity (at least $\Omega(2^{128})$ basic operations).

References

1. GOST R 34.12–2015: Information technology. cryptographic data security. block ciphers, 2015. English version; approved June 19, 2015; effective date 2016-01-01.
2. Alex Biryukov, Léo Perrin, and Aleksei Udovenko. Reverse-engineering the S-box of streebog, kuznyechik and STRIBOBr1. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology – EUROCRYPT 2016, Part I*, volume 9665 of *Lecture Notes in Computer Science*, pages 372–402. Springer, 2016.
3. Eli Biham and Adi Shamir. Differential cryptanalysis of DES-like cryptosystems. *Journal of Cryptology*, 4:3–72, 1991.
4. Lars R. Knudsen. Truncated and higher order differentials. In *Fast Software Encryption*, Lecture Notes in Computer Science. Springer, 1995.
5. Joan Daemen, Lars R. Knudsen, and Vincent Rijmen. The block cipher SQUARE. In *Fast Software Encryption*, Lecture Notes in Computer Science. Springer, 1997.
6. Lars R. Knudsen and David Wagner. Integral cryptanalysis. In *Fast Software Encryption*, Lecture Notes in Computer Science. Springer, 2002.
7. Riham AlTawy and Amr M. Youssef. A meet in the middle attack on reduced round kuznyechik. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E98-A(10):2194–2198, October 2015. Letter.
8. Riham AlTawy, Onur Duman, and Amr M. Youssef. Fault analysis of Kuznyechik. Cryptology ePrint Archive, Report 2015/347, 2015.
9. Alex Biryukov, Dmitry Khovratovich, and Léo Perrin. Multiset-algebraic cryptanalysis of reduced Kuznyechik, khazad and other ciphers with random linear layers. *IACR Transactions on Symmetric Cryptology*, (2):226–247, 2016.
10. Muhammad ElSheikh and Amr M. Youssef. On MILP-based automatic search for bit-based division property for ciphers with (large) linear layers. In Joonsang Baek and Sushmita Ruj, editors, *Information Security and Privacy: 26th Australasian Conference, ACISP 2021, Virtual Event, December 1–3, 2021, Proceedings*, volume 13083 of *Lecture Notes in Computer Science*, pages 111–131. Springer, 2021.
11. Pantelimon Stanica, Ranit Dutta, and Bimal Mandal. Extended c-differential distinguishers of full 9 and reduced-round kuznyechik cipher, no pre-whitening. Cryptology ePrint Archive, Paper 2025/1238, 2025.
12. Boykuziev Ilkhom, Angshuman Khan, Bakhtiyor Abdurakhimov, Rupayan Das, and Zarif Khudoykulov. Integral cryptanalysis: a new key determination technique for 3-phase kuznyechik encryption. *Engineering Research Express*, 5(3):035018, September 2023.

A Full type-level trace for one-round inversion

This appendix provides the complete type-level backward propagation used in Section 3.3. We describe the inversion of the linear layer L^{-1} as sixteen successive applications of R^{-1}, followed by the inverse S-box layer S^{-1}. As explained in the main text, key addition does not affect the type-level propagation because it adds a fixed constant to every element of the structure.

We start from the first-order structure

$$\begin{pmatrix} A & C & C & C \\ C & C & C & C \\ C & C & C & C \\ C & C & C & C \end{pmatrix}.$$

For readability we show the inverse sequence in the natural order: first apply S^{-1} (which preserves the $\mathbf{A}/\mathbf{C}$ pattern), then apply $R_1^{-1}, R_2^{-1}, \ldots, R_{16}^{-1}$.

Complete backward evolution

$$\begin{pmatrix} A & A & A & A \\ A & A & A & A \\ A & A & A & A \\ A & A & A & A \end{pmatrix} \xleftarrow{S^{-1}} \begin{pmatrix} A & A & A & A \\ A & A & A & A \\ A & A & A & A \\ A & A & A & A \end{pmatrix} \xleftarrow{R_1^{-1}} \begin{pmatrix} A & A & A & A \\ A & A & A & A \\ A & A & A & A \\ A & A & A & A \end{pmatrix}$$

$$\xleftarrow{R_2^{-1}} \begin{pmatrix} A & A & A & A \\ A & A & A & A \\ A & A & A & A \\ A & A & A & C \end{pmatrix} \xleftarrow{R_3^{-1}} \begin{pmatrix} A & A & A & A \\ A & A & A & A \\ A & A & A & A \\ A & A & C & C \end{pmatrix} \xleftarrow{R_4^{-1}} \begin{pmatrix} A & A & A & A \\ A & A & A & A \\ A & A & A & A \\ A & C & C & C \end{pmatrix}$$

$$\xleftarrow{R_5^{-1}} \begin{pmatrix} A & A & A & A \\ A & A & A & A \\ A & A & A & A \\ C & C & C & C \end{pmatrix} \xleftarrow{R_6^{-1}} \begin{pmatrix} A & A & A & A \\ A & A & A & A \\ A & A & A & C \\ C & C & C & C \end{pmatrix} \xleftarrow{R_7^{-1}} \begin{pmatrix} A & A & A & A \\ A & A & A & A \\ A & A & C & C \\ C & C & C & C \end{pmatrix}$$

$$\xleftarrow{R_8^{-1}} \begin{pmatrix} A & A & A & A \\ A & A & A & A \\ A & C & C & C \\ C & C & C & C \end{pmatrix} \xleftarrow{R_9^{-1}} \begin{pmatrix} A & A & A & A \\ A & A & A & A \\ C & C & C & C \\ C & C & C & C \end{pmatrix} \xleftarrow{R_{10}^{-1}} \begin{pmatrix} A & A & A & A \\ A & A & A & C \\ C & C & C & C \\ C & C & C & C \end{pmatrix}$$

$$\xleftarrow{R_{11}^{-1}} \begin{pmatrix} A & A & A & A \\ A & A & C & C \\ C & C & C & C \\ C & C & C & C \end{pmatrix} \xleftarrow{R_{12}^{-1}} \begin{pmatrix} A & A & A & A \\ A & C & C & C \\ C & C & C & C \\ C & C & C & C \end{pmatrix} \xleftarrow{R_{13}^{-1}} \begin{pmatrix} A & A & A & A \\ C & C & C & C \\ C & C & C & C \\ C & C & C & C \end{pmatrix}$$

$$\xleftarrow{R_{14}^{-1}} \begin{pmatrix} A & A & A & C \\ C & C & C & C \\ C & C & C & C \\ C & C & C & C \end{pmatrix} \xleftarrow{R_{15}^{-1}} \begin{pmatrix} A & A & C & C \\ C & C & C & C \\ C & C & C & C \\ C & C & C & C \end{pmatrix} \xleftarrow{R_{16}^{-1}} \begin{pmatrix} A & C & C & C \\ C & C & C & C \\ C & C & C & C \\ C & C & C & C \end{pmatrix}$$

The above trace matches the type-level behavior observed in our experimental verification code and supports the one-round preimage construction used to extend the 2-round distinguisher to three rounds.

B Full type-level propagation through the *R*-steps

This appendix lists the detailed type-level evolution through the intermediate steps S and the sixteen R transformations that compose the linear layer L, i.e., $L = R_{16} \circ \cdots \circ R_1$. The traces are shown for the first and second rounds starting from the first-order input structure (A, C^{15}) in row-major 4×4 layout.

B.1 Round 1 transition

$$\begin{pmatrix} A & C & C & C \\ C & C & C & C \\ C & C & C & C \\ C & C & C & C \end{pmatrix} \xrightarrow{S} \begin{pmatrix} A & C & C & C \\ C & C & C & C \\ C & C & C & C \\ C & C & C & C \end{pmatrix} \xrightarrow{R_1} \begin{pmatrix} A & A & C & C \\ C & C & C & C \\ C & C & C & C \\ C & C & C & C \end{pmatrix}$$

$$\xrightarrow{R_2} \begin{pmatrix} A & A & A & C \\ C & C & C & C \\ C & C & C & C \\ C & C & C & C \end{pmatrix} \xrightarrow{R_3} \begin{pmatrix} A & A & A & A \\ C & C & C & C \\ C & C & C & C \\ C & C & C & C \end{pmatrix} \xrightarrow{R_4} \begin{pmatrix} A & A & A & A \\ A & C & C & C \\ C & C & C & C \\ C & C & C & C \end{pmatrix}$$

$$\xrightarrow{R_5} \begin{pmatrix} A & A & A & A \\ A & A & C & C \\ C & C & C & C \\ C & C & C & C \end{pmatrix} \xrightarrow{R_6} \begin{pmatrix} A & A & A & A \\ A & A & A & C \\ C & C & C & C \\ C & C & C & C \end{pmatrix} \xrightarrow{R_7} \begin{pmatrix} A & A & A & A \\ A & A & A & A \\ C & C & C & C \\ C & C & C & C \end{pmatrix}$$

$$\xrightarrow{R_8} \begin{pmatrix} A & A & A & A \\ A & A & A & A \\ A & C & C & C \\ C & A & C & C \end{pmatrix} \xrightarrow{R_9} \begin{pmatrix} A & A & A & A \\ A & A & A & A \\ A & A & C & C \\ C & C & C & C \end{pmatrix} \xrightarrow{R_{10}} \begin{pmatrix} A & A & A & A \\ A & A & A & A \\ A & A & A & C \\ C & C & C & C \end{pmatrix}$$

$$\xrightarrow{R_{11}} \begin{pmatrix} A & A & A & A \\ A & A & A & A \\ A & A & A & A \\ C & C & C & C \end{pmatrix} \xrightarrow{R_{12}} \begin{pmatrix} A & A & A & A \\ A & A & A & A \\ A & A & A & A \\ A & C & C & C \end{pmatrix} \xrightarrow{R_{13}} \begin{pmatrix} A & A & A & A \\ A & A & A & A \\ A & A & A & A \\ A & A & C & C \end{pmatrix}$$

$$\xrightarrow{R_{14}} \begin{pmatrix} A & A & A & A \\ A & A & A & A \\ A & A & A & A \\ A & A & A & C \end{pmatrix} \xrightarrow{R_{15}} \begin{pmatrix} A & A & A & A \\ A & A & A & A \\ A & A & A & A \\ A & A & A & A \end{pmatrix} \xrightarrow{R_{16}} \begin{pmatrix} A & A & A & A \\ A & A & A & A \\ A & A & A & A \\ A & A & A & A \end{pmatrix}$$

B.2 Round 2 transition

$$\begin{pmatrix} A & A & A & A \\ A & A & A & A \\ A & A & A & A \\ A & A & A & A \end{pmatrix} \xrightarrow{S} \begin{pmatrix} A & A & A & A \\ A & A & A & A \\ A & A & A & A \\ A & A & A & A \end{pmatrix} \xrightarrow{R_1} \begin{pmatrix} B & A & A & A \\ A & A & A & A \\ A & A & A & A \\ A & A & A & A \end{pmatrix}$$

$$\xrightarrow{R_2} \begin{pmatrix} B & B & A & A \\ A & A & A & A \\ A & A & A & A \\ A & A & A & A \end{pmatrix} \xrightarrow{R_3} \begin{pmatrix} B & B & B & A \\ A & A & A & A \\ A & A & A & A \\ A & A & A & A \end{pmatrix} \xrightarrow{R_4} \begin{pmatrix} B & B & B & B \\ A & A & A & A \\ A & A & A & A \\ A & A & A & A \end{pmatrix}$$

$$\xrightarrow{R_5} \begin{pmatrix} B & B & B & B \\ B & A & A & A \\ A & A & A & A \\ A & A & A & A \end{pmatrix} \xrightarrow{R_6} \begin{pmatrix} B & B & B & B \\ B & B & A & A \\ A & A & A & A \\ A & A & A & A \end{pmatrix} \xrightarrow{R_7} \begin{pmatrix} B & B & B & B \\ B & B & B & A \\ A & A & A & A \\ A & A & A & A \end{pmatrix}$$

$$\xrightarrow{R_8} \begin{pmatrix} B & B & B & B \\ B & B & B & B \\ A & A & A & A \\ A & A & A & A \end{pmatrix} \xrightarrow{R_9} \begin{pmatrix} B & B & B & B \\ B & B & B & B \\ B & A & A & A \\ A & A & A & A \end{pmatrix} \xrightarrow{R_{10}} \begin{pmatrix} B & B & B & B \\ B & B & B & B \\ B & B & A & A \\ A & A & A & A \end{pmatrix}$$

$$\xrightarrow{R_{11}} \begin{pmatrix} B & B & B & B \\ B & B & B & B \\ B & B & B & A \\ A & A & A & A \end{pmatrix} \xrightarrow{R_{12}} \begin{pmatrix} B & B & B & B \\ B & B & B & B \\ B & B & B & B \\ A & A & A & A \end{pmatrix} \xrightarrow{R_{13}} \begin{pmatrix} B & B & B & B \\ B & B & B & B \\ B & B & B & B \\ B & A & A & A \end{pmatrix}$$

$$\xrightarrow{R_{14}} \begin{pmatrix} B & B & B & B \\ B & B & B & B \\ B & B & B & B \\ B & B & A & A \end{pmatrix} \xrightarrow{R_{15}} \begin{pmatrix} B & B & B & B \\ B & B & B & B \\ B & B & B & B \\ B & B & B & A \end{pmatrix} \xrightarrow{R_{16}} \begin{pmatrix} B & B & B & B \\ B & B & B & B \\ B & B & B & B \\ B & B & B & B \end{pmatrix}$$

B.3 Round 3 onwards

In the third round of the Kuznyechik cipher, the previously observed *BALANCED* property further degrades under the successive application of the substitution layer S and the linear transformations (R_i). Consequently, the state transitions into a *MIXED* (M) property, where no exploitable integral structure remains and the state behaves in a fully diffused manner.

$$\begin{pmatrix} B & B & B & B \\ B & B & B & B \\ B & B & B & B \\ B & B & B & B \end{pmatrix} \xrightarrow{\text{4th round}} \begin{pmatrix} M & M & M & M \\ M & M & M & M \\ M & M & M & M \\ M & M & M & M \end{pmatrix} \overset{\text{9th round}}{\cdots} \begin{pmatrix} M & M & M & M \\ M & M & M & M \\ M & M & M & M \\ M & M & M & M \end{pmatrix}$$