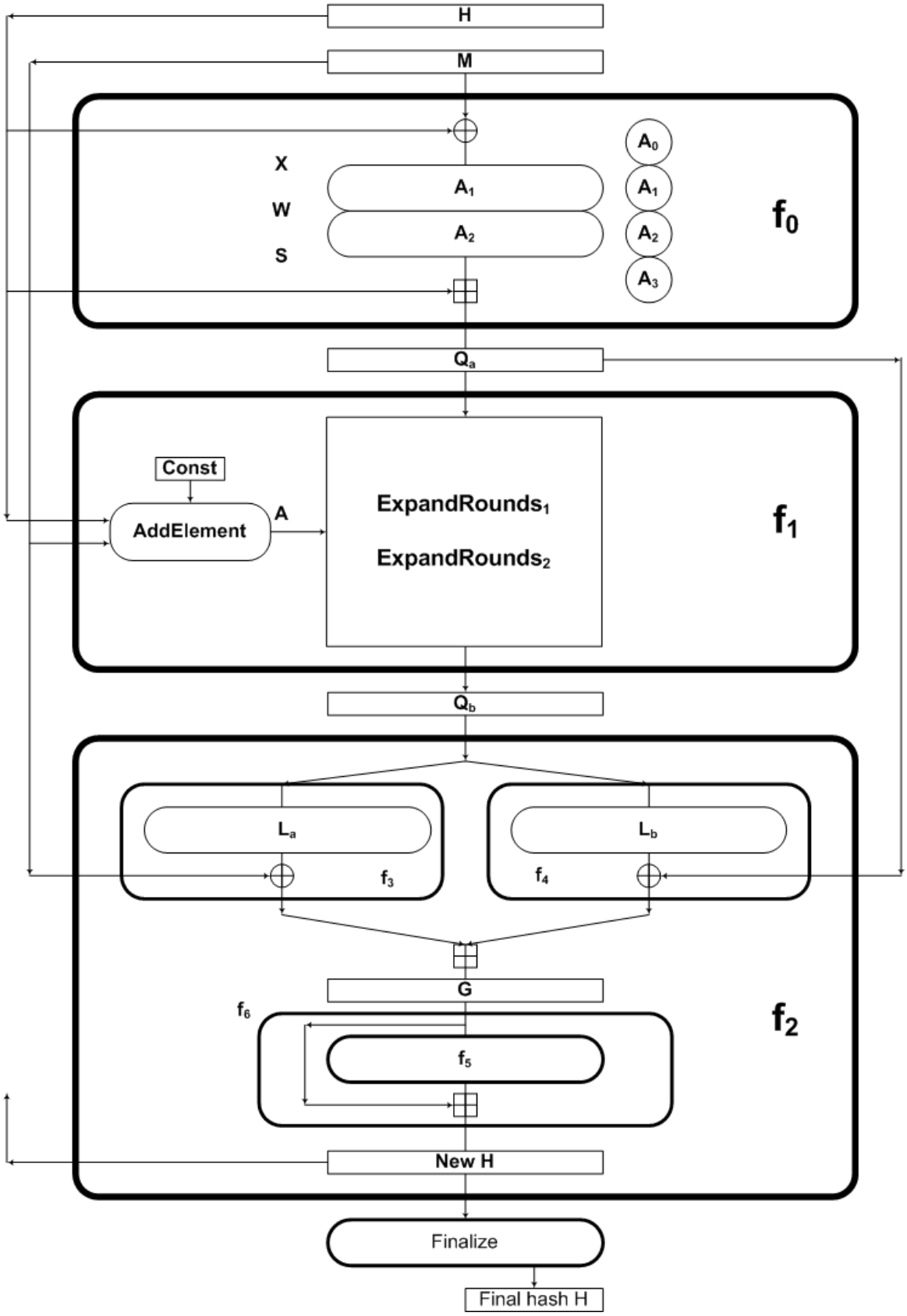
A. Poznámka k lineárním aproximacím kryptografické hašovací funkce BLUE MIDNIGHT WISH

Vlastimil Klíma, nezávislý kryptolog, (v.klima@volny.cz)
Petr Sušil, PhD student, EPFL, (susil.petr@gmail.com)

Abstrakt. Hašovací funkce BLUE MIDNIGHT WISH (BMW) je nejrychlejší ze 14 kandidátů v 2. kole soutěže SHA-3 [1]. Na začátku tohoto kola byli autoři kandidátů vyzváni, aby před 15. zářím upravili své algoritmy (tzv. tweak). V tomto příspěvku se budeme zabývat tedy nejnovější "tweakovanou" verzí BMW [3]. Algoritmus BMW je typu AXR, protože používá pouze operace ADD (sub), XOR a ROT (shift). Když operaci ADD nahradíme XOR, obdržíme BMW_{lin}, což je afinní transformace. V příspěvku uvažujeme pouze funkci BMW_{lin} a její stavební bloky. Tyto afinní transformace mohou být reprezentovány lineární maticí a konstantním vektorem. Zjistili jsme, že všechny matice vyšších stavebních bloků BMW_{lin} mají plnou hodnost nebo hodnost blízkou k ní. Také jsme zkoumali strukturu těchto matic. Matice dílčích stavebních bloků mají očekávanou nenáhodnou strukturu, zatímco matice vyšších bloků mají strukturu již náhodnou. Ukážeme také matice pro různé hodnoty $ExpandRounds_1$ (pro hodnoty mezi 0 a 16). Jejich zvyšování vede k větší náhodnosti matic, což bylo návrháři zamýšleno. Tato pozorování platí pro obě verze $BMW256_{lin}$ a $BMW512_{lin}$. V této lineární analýze jsme nenašli žádnou užitečnou vlastnost, která by pomohla kryptoanalýze, ani jsme nenašli žádnou slabost BMW. Studium dílčích bloků bude následovat.



Obr.1: Schéma (tweakované verze) BMW [3]

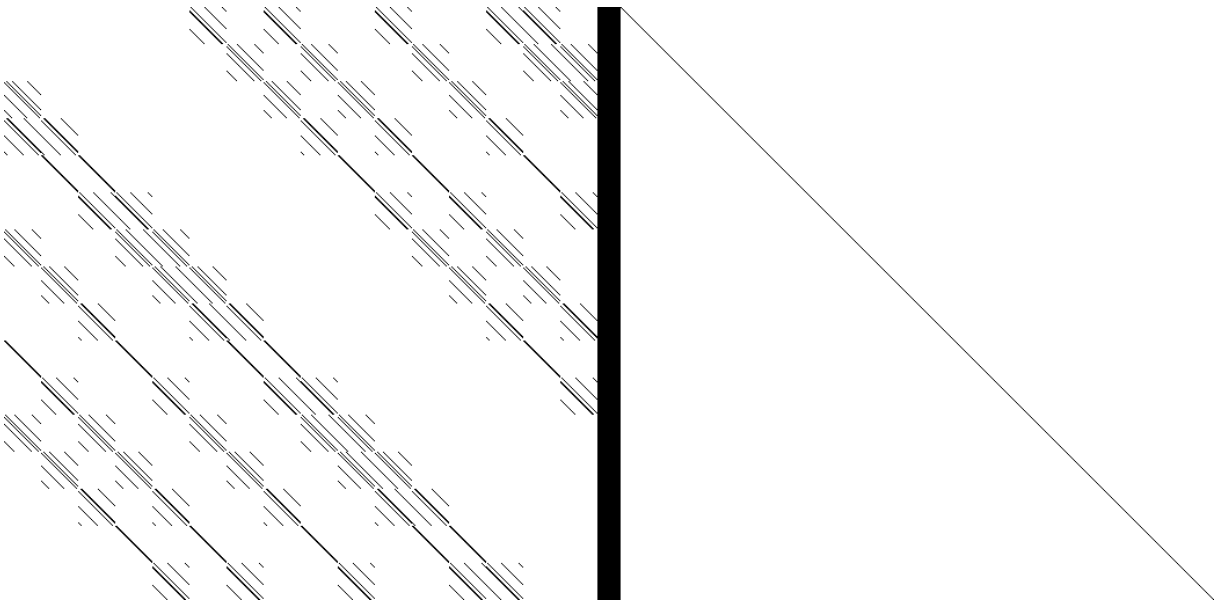
Úvod

V příspěvku uvažujeme pouze funkci BMW_{lin} a její stavební bloky. Tyto afinní transformace mohou být reprezentovány maticemi. Uvádíme hodnost a strukturu těchto matic. Hlavní bloky BMW jsou f_0, f_1, f_2. Vytváří tři meziproměnné Q_a, Q_b, G a průběžnou haš H (viz obr. 1). Všechny čtyři proměnné jsou 16*w - bitová slova (w = 32/64 pro BMW256/512). Tyto proměnné závisí pouze na vstupním bloku M a na staré průběžné hašovací hodnotě H. Právě tyto závislosti budou ukázány v maticích. Povšimněte si hodnosti a struktury každé matice. Některé matice budou ukázány pro několik různých hodnot bezpečnostního parametru $ExpandRounds_1$. Připomeňme, že po $ExpandRounds_1$ rundách typu $expand_1$ následuje 16 - $ExpandRounds_2$ rund typu $expand_2$ [3]. Pro rozsáhlost tohoto příspěvku předpokládáme, že čtenář je seznámen se základním popisem BMW v [3]. Také používáme jednoduché označení H, ale kdykoli by mohlo dojít k nepochopení, která z hodnot to je, odlišujeme je jako oldH a newH (viz obr. 1). Naše pozorování platí pro obě verze obě verze $BMW256_{lin}$ a $BMW512_{lin}$, proto z důvodu jednoduchosti prezentujeme výsledky pouze pro $BMW256_{lin}$.

Na následujícím obrázku vidíme lineární závislost mezi M a Q_a. Jsou zde dvě matice typu 512 x 512 (oddělené černým pruhem), což označujeme jako (M, Q_a). Sloupce reprezentují 512 proměnných (bitů) proměnné M na levé straně a 512 proměnných (bitů) proměnné Q_a na pravé straně. Řádky pak reprezentují lineární závislosti mezi bity na levé straně (M) a bity na pravé straně (Q_a). Protože matice na pravé straně je identická, dává nám to přímou závislost bitů Q_a na bitech proměnné M. Každý řádek může být také zapsán jako lineární rovnice

$$\bigoplus_{i \in Left} x_i = \bigoplus_{j \in Right} y_j.$$

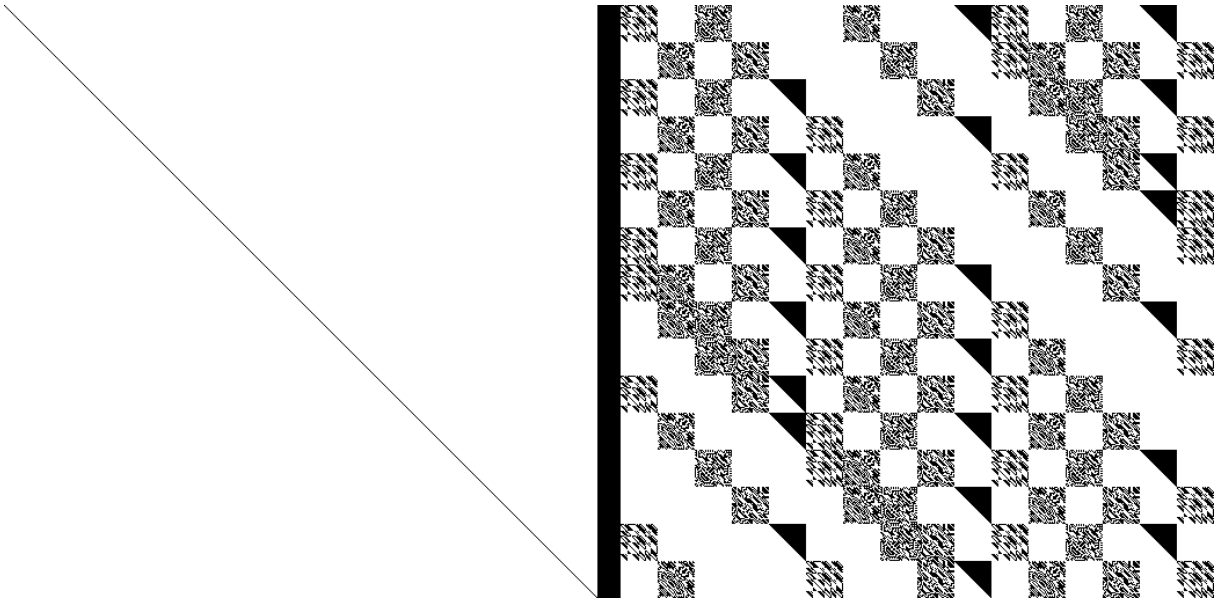
Indexy bitů, které se objevují v rovnici, jsou na obrázku 2 označeny jako černé body. Tím zde mj. můžeme vidět závislost prvního slova Q_a na pěti slovech M, což vyplývá z definice transformace A_1 ve [3].



Obr.2: Příklad závislosti mezi proměnnými (bity) M a Q_a, což označujeme jako (M, Q_a).

Když bychom chtěli vidět závislost M na Q_a, potřebovali bychom inverzní matici. Jak víme z lineární algebry, xorováním jednoho řádku s jiným nebo jejich výměna nemění hodnost matice. S využitím těchto dvou operací v našem eliminačním algoritmu transformujeme dvojici matic (A, I) na (I´, A´), kde I je identická matice a I´ je identická, pokud matice A má plnou hodnost. Je-li I´ identita, pak A´ ukazuje závislosti M na Q_a, viz následující obrázek. Je-li A singulární, pak I´ bude mít neprázdný sloupec (sloupce) nad diagonálou. Poznamenejme, že proměnná, která odpovídá tomuto neprázdnému sloupci není nezávislá a vystupuje pouze v lineární kombinaci s dalšími proměnnými (bity) v matici, viz například obrázek 4 (výřez obrázku pro hodnost = MAX - 1).

V následujícím uvádíme pouze nejzajímavější závislosti.

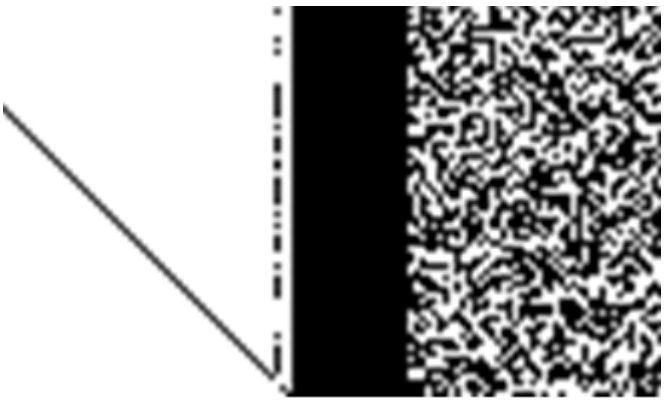


Obr.3: (M, Q_a) - závislost mezi M a Q_a

Elimination algorithm - transformation of (A, I) to (I′, A′):

```
Input:  boolean matrix A[LEN][LEN]
Output: pair of boolean matrices (I',A')
var boolean matrix I[LEN][LEN], where I[i][j]=1 if i==j and I[i][j]=0 otherwise

for i=1 to LEN
begin
      if A[i][i] then
        for all j != i and A[j][i] //add line i to line j in (A, I)
           for all k
               A[j][k] = A[j][k] + A[i][k] and I[j][k] = I[j][k] + I[i][k];
      else
        for all j > i
           if A[j][i]    //switch line j and i in (A, I)
              for all k
                  switch values A[j][k] and A[i][k] and I[j][k] and I[i][k]
              go to begin (without increasing i) or go to end (if i = LEN)
end
output (A,I)
```



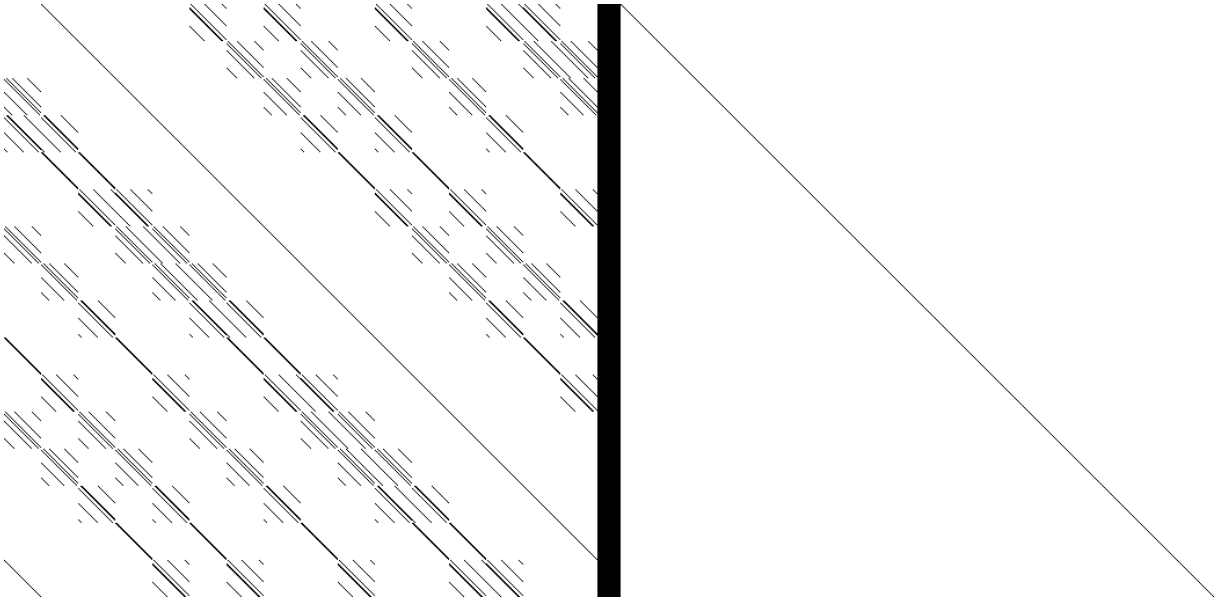
Obr.4: Výřez závislosti u matice, která má hodnost o 1 nižší než plnou

Závislost Qa na M

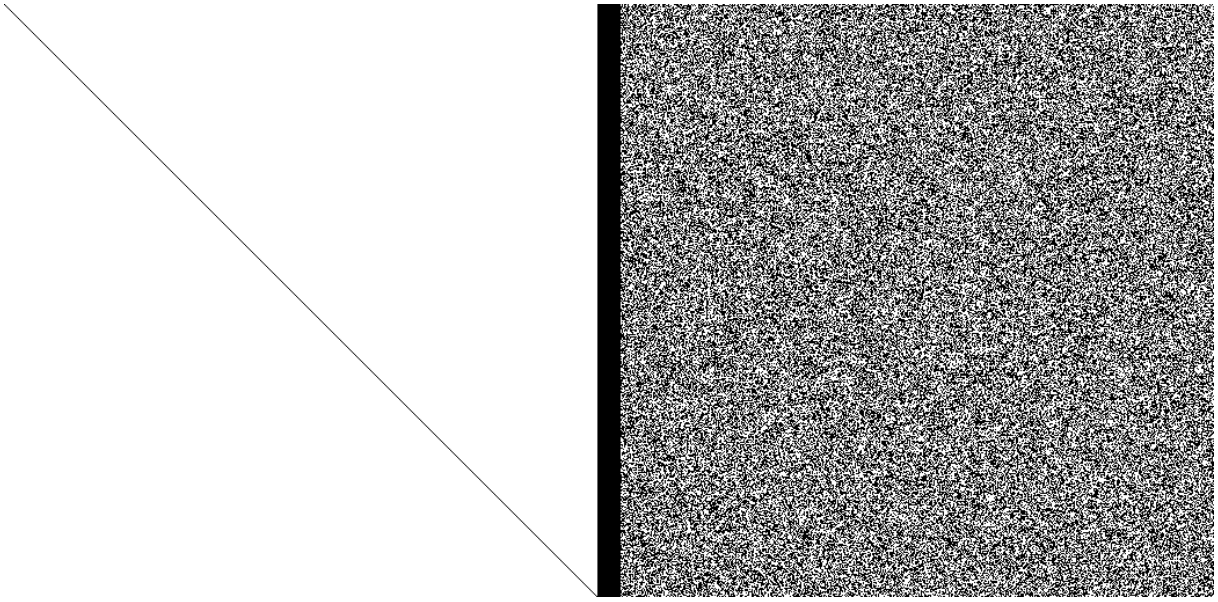
Závislost Q_a na M vidíme na obrázcích 2 a 3. Poznamenejme, že matice nezávisí na hodnotě ExpandRounds1, protože Q_a je vytvořena až ve funkci f_1. Hodnost matice (M, Q_a) je rank(M, Q_a) = 512, tj. plná.

Závislost Qa na H

Také hodnost matice (H, Q_a) je plná, a protože zobrazuje závislosti v bloku f_0, ještě nezávisí na hodnotě $ExpandRounds_1$.



Obr.5: (H, Q_a)



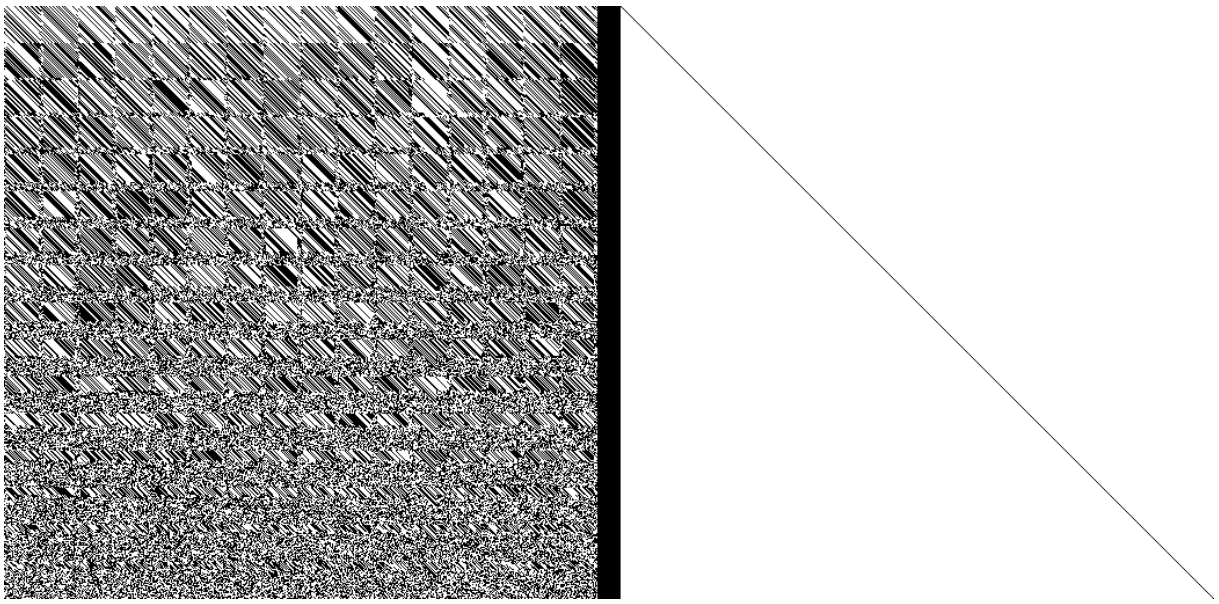
Obr.6: (H, Q_a)

Závislost Qb na M

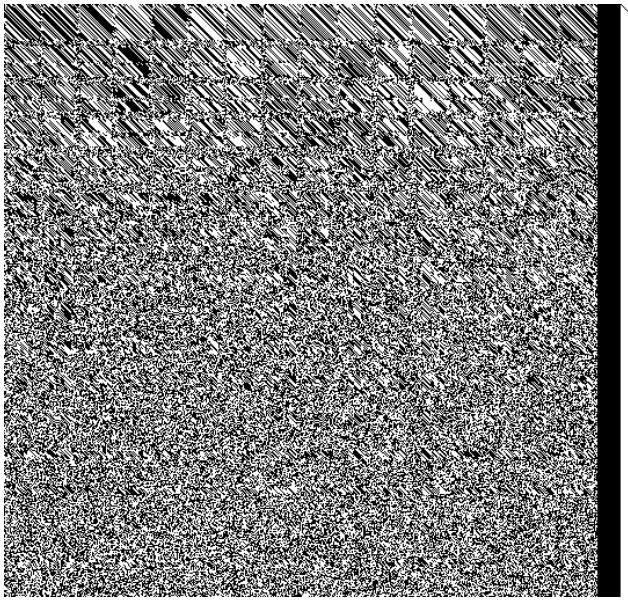
Zde můžeme poprvé vidět, že matice nemají plnou hodnost a že s růstem $ExpandRounds_1$ mají tendenci se znáhodňovat. Protože ostatní závislosti jsou podobné, v dalším uvádíme pouze hodnosti všech matic a ukázky některých.

$ExpandRounds_1$	0	1	2	3	4	5	6	7	8
$Rank(M, Q_b)$	511	512	512	510	511	512	512	511	510

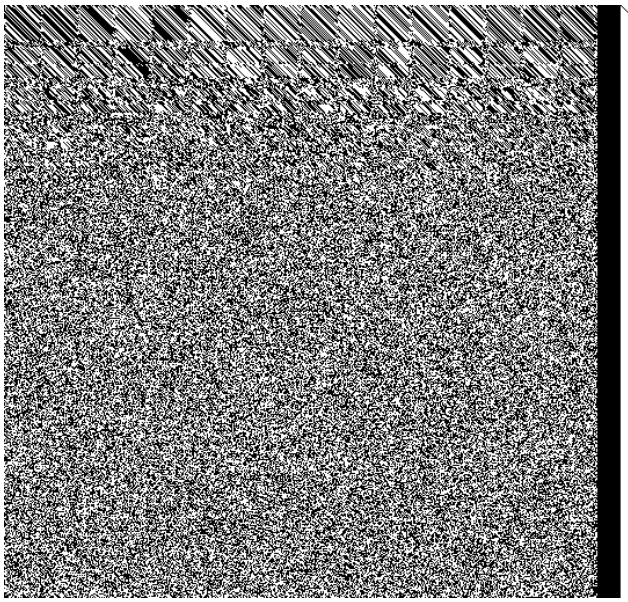
$ExpandRounds_1$	9	10	11	12	13	14	15	16
$Rank(M, Q_b)$	511	511	512	512	510	512	511	510



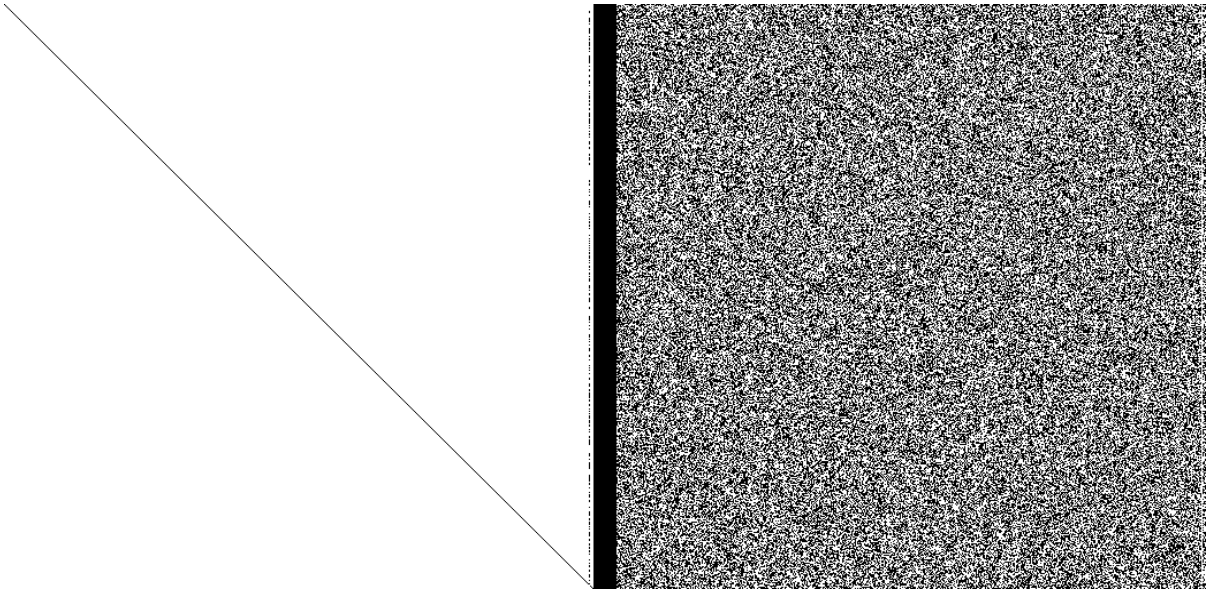
Obr.7: (M, Q_b), ExpandRounds1 = 0



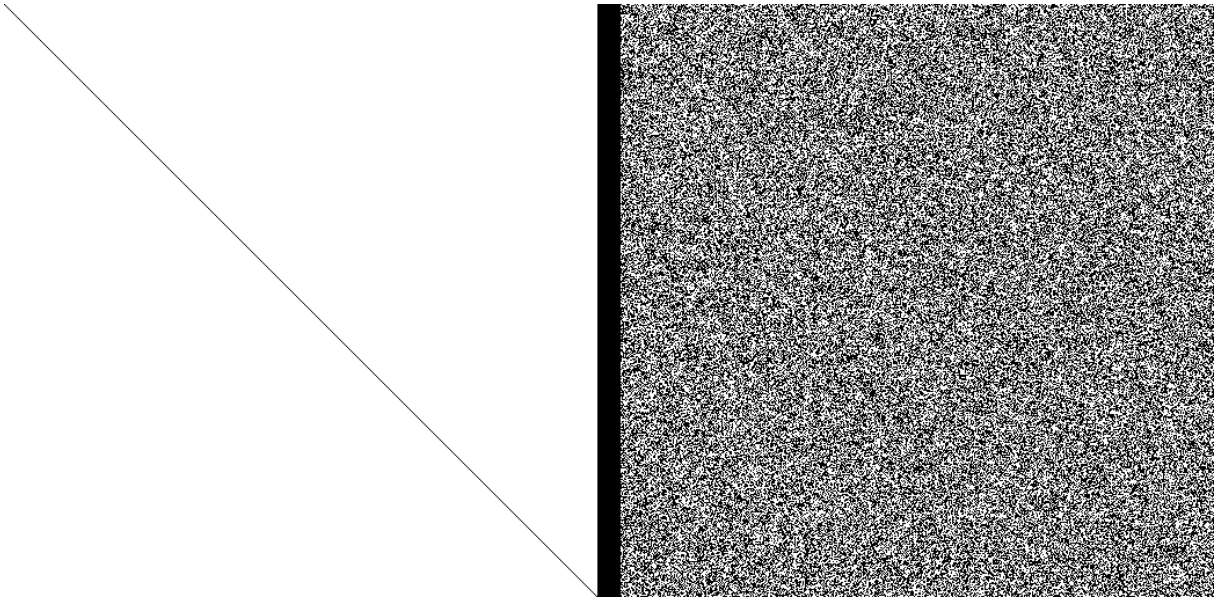
Obr.8: (M, Q_b), ExpandRounds1 = 2



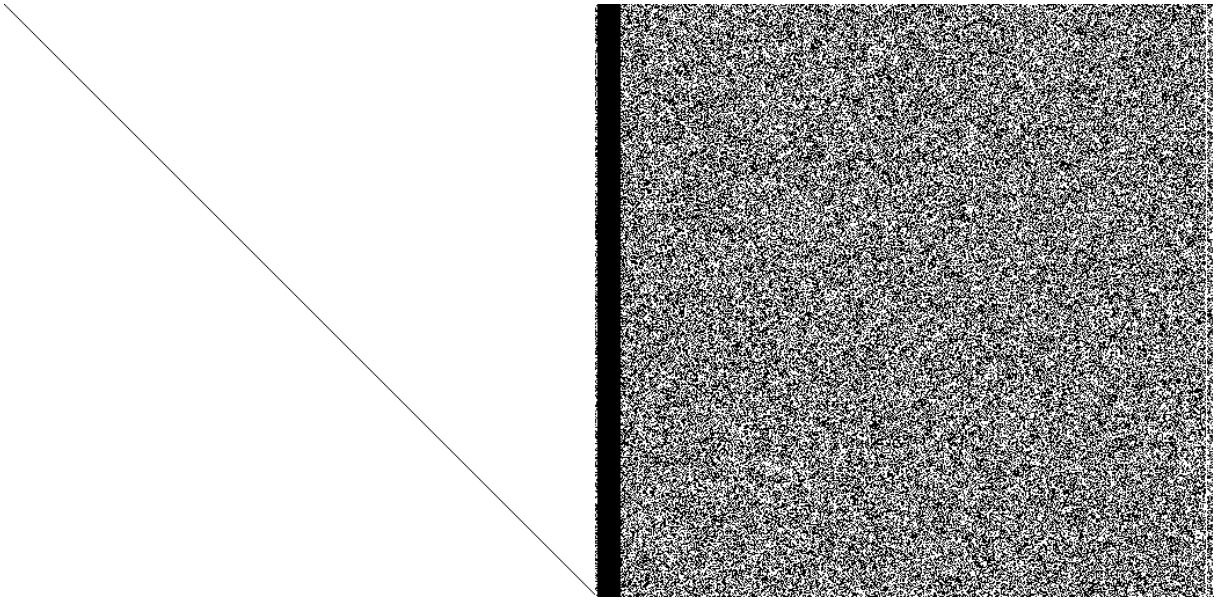
Obr.9: (M, Q_b), ExpandRounds1 = 16



Obr.10: (M, Q_b), ExpandRounds1 = 0



Obr.11: (M, Q_b), ExpandRounds1 = 2

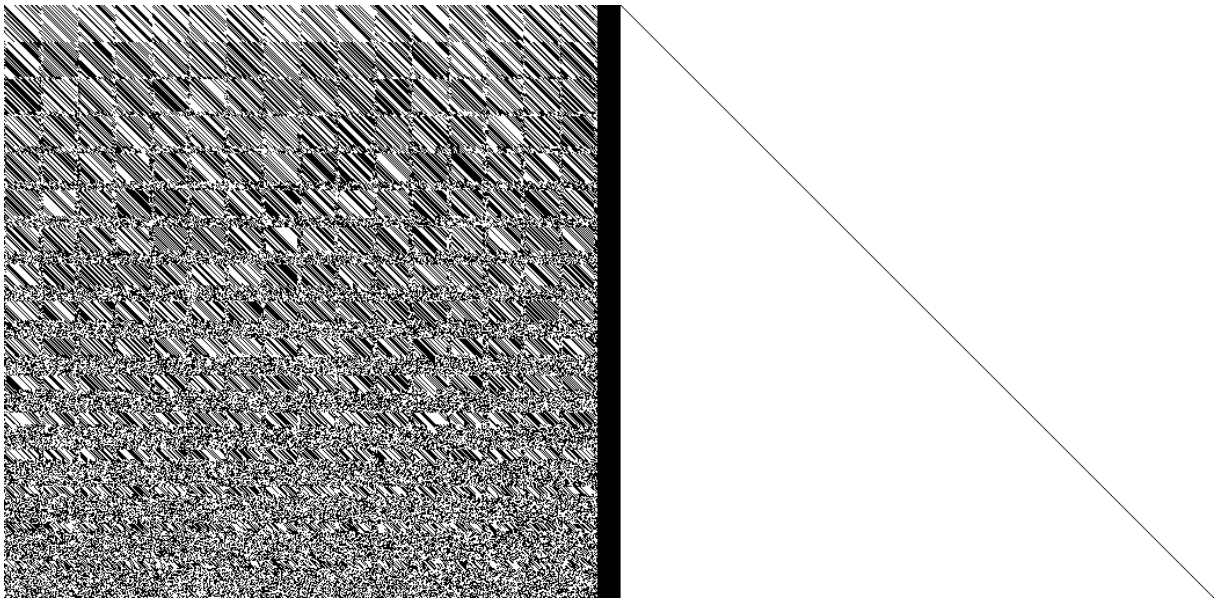


Obr.12: (M, Q_b), ExpandRounds1 = 16

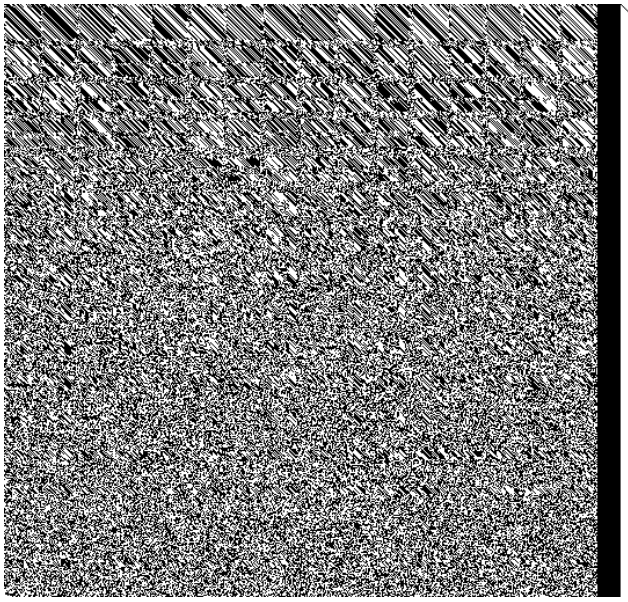
Závislost Qb na oldH

$ExpandRounds_1$	0	1	2	3	4	5	6	7	8
$Rank(oldH, Q_b)$	512	510	509	511	511	511	511	510	512

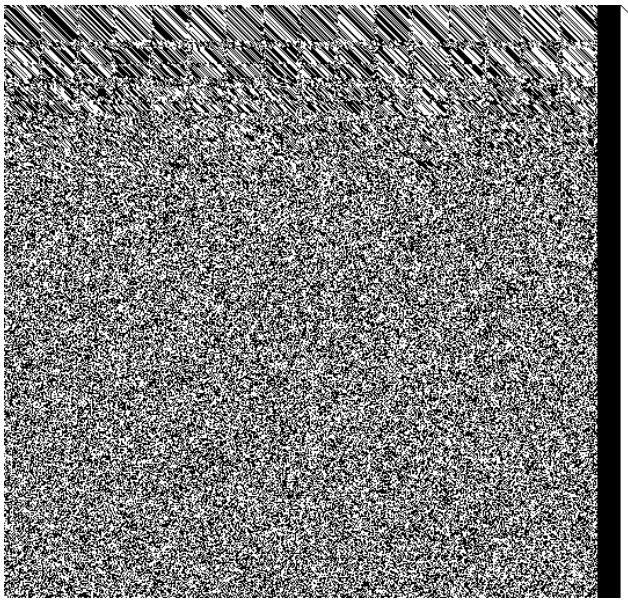
$ExpandRounds_1$	9	10	11	12	13	14	15	16
$Rank(oldH, Q_b)$	511	511	512	510	511	512	511	511



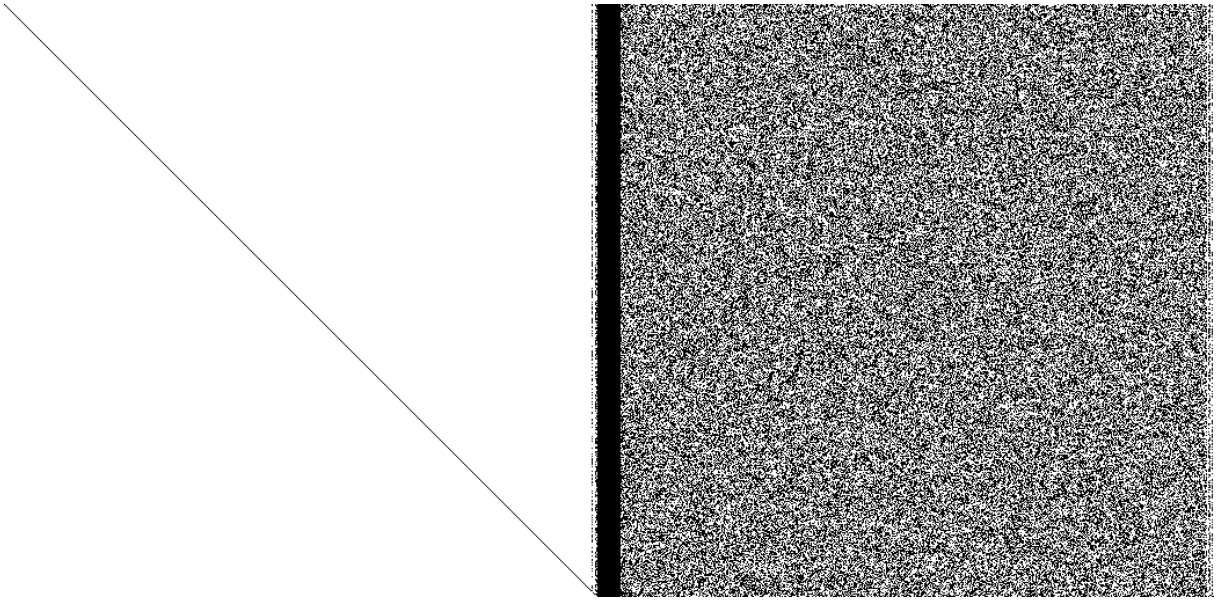
Obr.13: (oldH, Q_b), ExpandRounds1 = 0



Obr.14: (oldH, Q_b), ExpandRounds1 = 2



Obr.15: (oldH, Q_b), ExpandRounds1 = 16

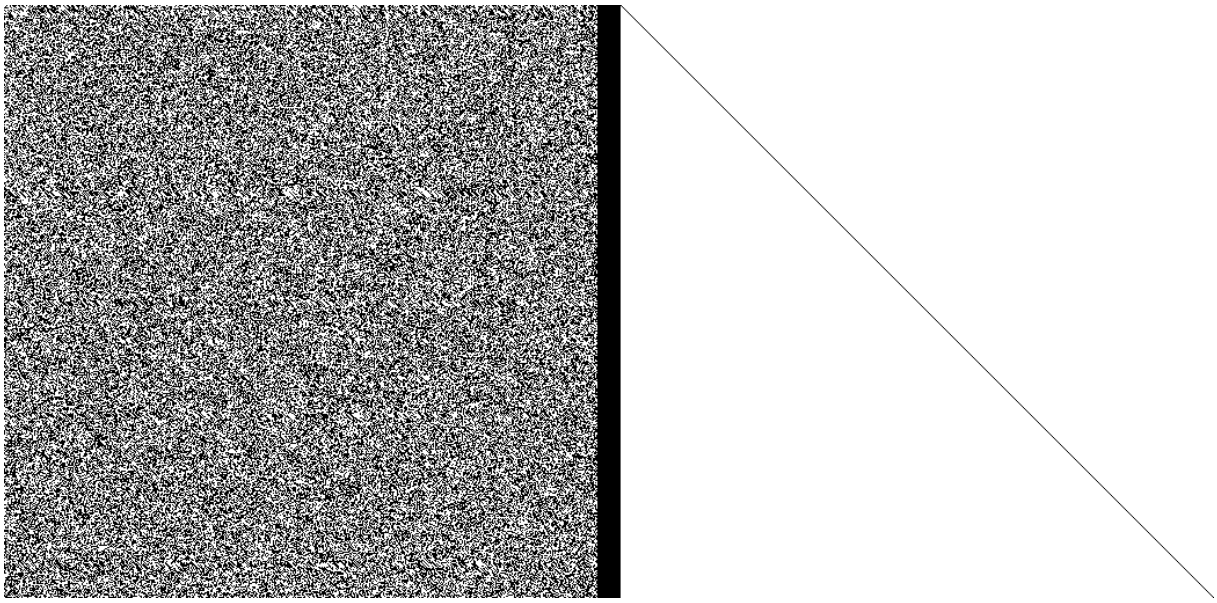


Obr.16: (oldH, Q_b), ExpandRounds1 = 2

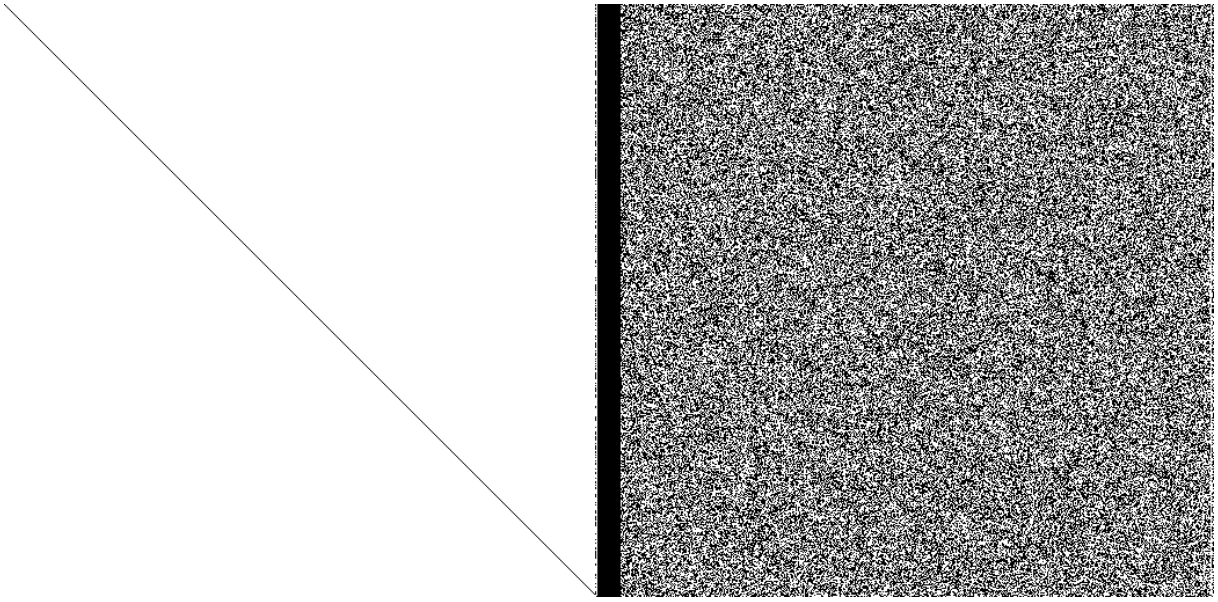
Závislost G na M

$ExpandRounds_1$	0	1	2	3	4	5	6	7	8
$Rank(M,G)$	512	510	511	512	510	511	510	512	511

$ExpandRounds_1$	9	10	11	12	13	14	15	16
$Rank(M,G)$	510	510	510	511	512	512	510	511



Obr.17: (M, G), ExpandRounds1 = 2

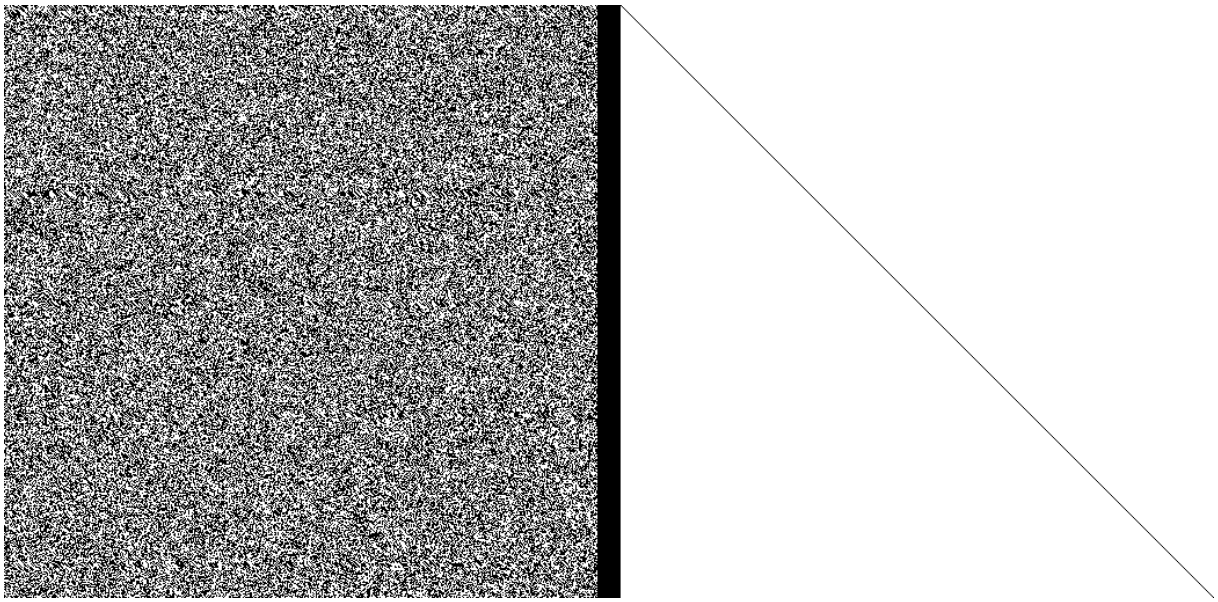


Obr.18: (M, G), ExpandRounds1 = 2

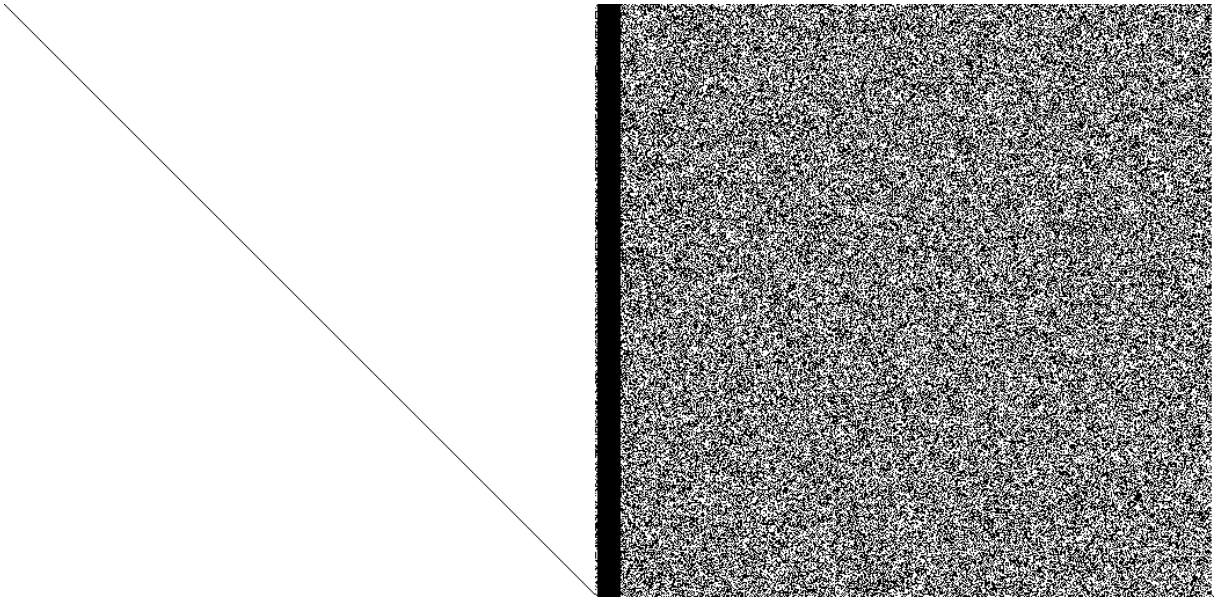
Závislost G na oldH

$ExpandRounds_1$	0	1	2	3	4	5	6	7	8
$Rank(oldH, G)$	511	511	510	511	511	511	512	511	511

$ExpandRounds_1$	9	10	11	12	13	14	15	16
$Rank(oldH, G)$	512	510	510	511	512	512	511	511



Obr.19: (oldH, G), ExpandRounds1 = 2

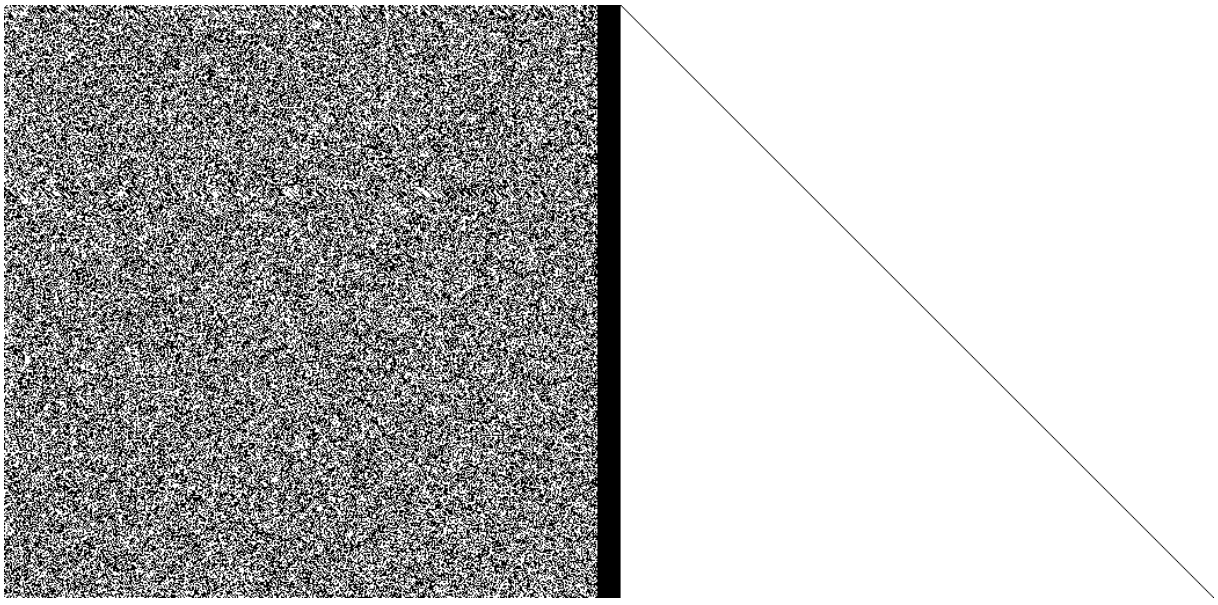


Obr.20: (oldH, G), ExpandRounds1 = 2

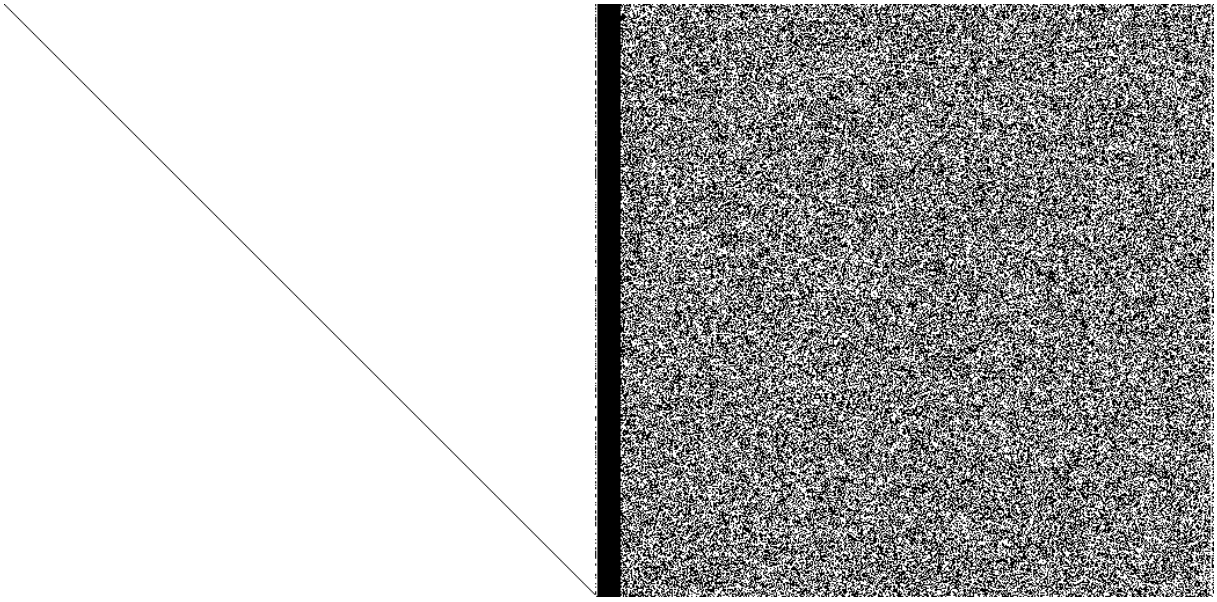
Závislost newH na M

$ExpandRounds_1$	0	1	2	3	4	5	6	7	8
$Rank(oldH, G)$	512	510	511	512	510	511	510	512	511

$ExpandRounds_1$	9	10	11	12	13	14	15	16
$Rank(oldH, G)$	510	510	510	511	512	512	510	511



Obr.21: (M, newH), ExpandRounds1 = 2

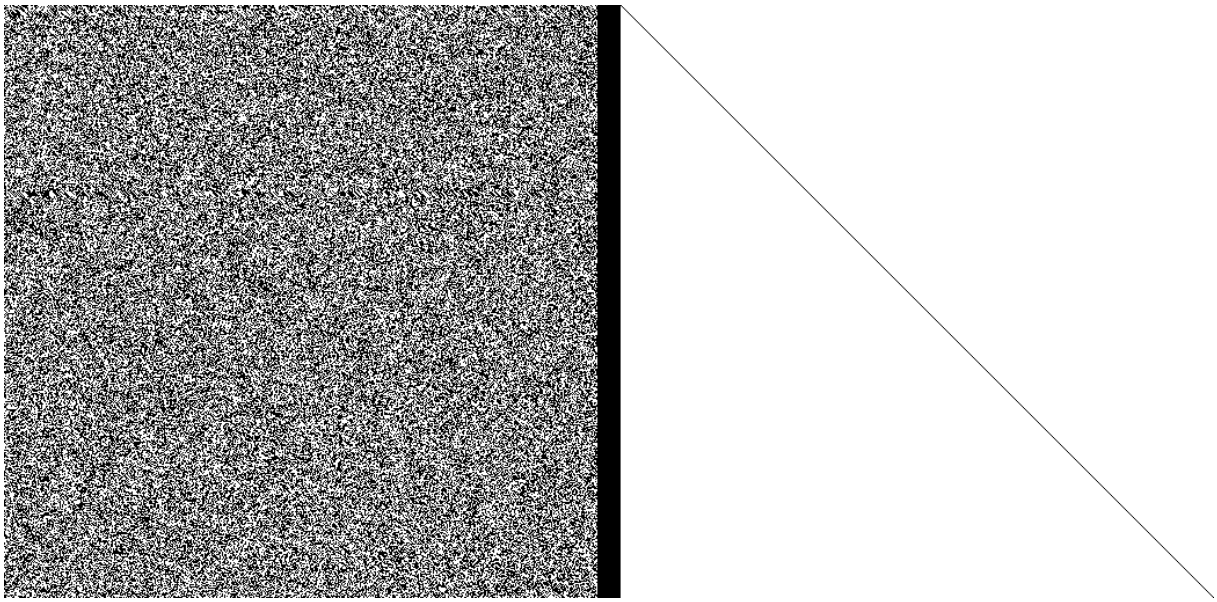


Obr.22: (M, newH), ExpandRounds1 = 2

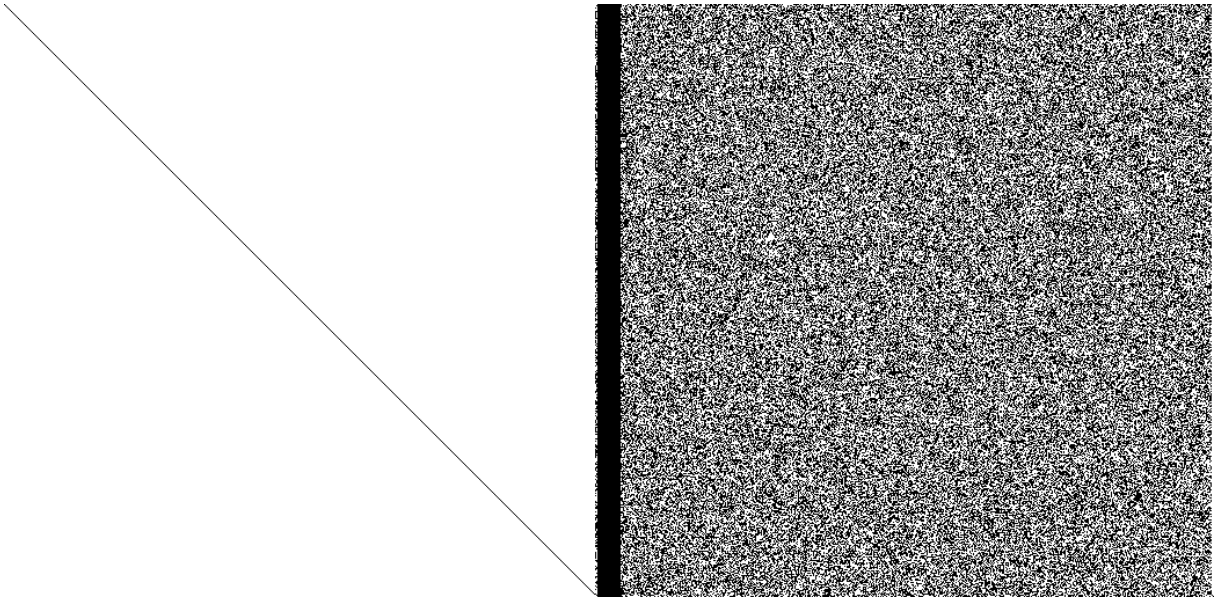
Závislost newH na oldH

$ExpandRounds_1$	0	1	2	3	4	5	6	7	8
$Rank(oldH, newH)$	511	511	510	511	511	511	512	511	511

$ExpandRounds_1$	9	10	11	12	13	14	15	16
$Rank(oldH, newH)$	512	510	510	511	512	512	511	511



Obr.23: (oldH, newH), ExpandRounds1 = 2



Obr.24: (oldH, newH), ExpandRounds1 = 2

Závěr

Výsledky prezentované v tomto příspěvku jsou pouze malou částí analýz, které byly provedeny. Jak jste si povšimli, jsou zde zkoumány pouze závislosti na proměnných M a H. Avšak byly analyzovány i některé dílčí i vyšší stavební bloky zvlášť (tj. i s jinými vstupy), což dává hlubší pohled do struktury BMW. Výsledky úplnější analýzy budou následovat. V tomto příspěvku jsme viděli, že všechny závislosti vytvářely matice s plnou hodností nebo jí blízkou. Také struktura matic byla očekávaná: matice dílčích bloků neměly náhodnou strukturu, ale splňovaly požadavky rychlého míchání proměnných (bitů). Matice vyšších stavebních bloků měly plnou nebo skoro plnou hodnost a náhodnou strukturu. Viděli jsme také, že je-li požadováno (a možné) zvyšovat náhodnost, může to být uděláno zvyšováním počtu rund $ExpandRounds_1$. Tato pozorování platí pro obě dvě hlavní verze BMW256 a BMW512.

Literatura

[1] Announcing Request for Candidate Algorithm Nominations for a New Cryptographic Hash Algorithm (SHA-3) Family, 2007, NIST, http://csrc.nist.gov/groups/ST/hash/index.html.
[2] Danilo Gligoroski, Vlastimil Klima, On BLUE MIDNIGHT WISH decomposition, to be published
[3] Danilo Gligoroski, Vlastimil Klima, Svein Johan Knapskog, Mohamed El-Hadedy, Jørn Amundsen, Stig Frode Mjølsnes: Cryptographic Hash Function Blue Midnight Wish, September 2009, http://people.item.ntnu.no/~danilog/Hash/BMW-SecondRound/Supporting Documentation/BlueMidnightWishDocumentation.pdf