ივანე ჯავახიშვილის სახელობის თბილისის სახელმწიფო უნივერსიტეტი

ირაკლი მანაგაძე

ციფრული კომუნიკაციები: შეცდომების მაკორექტირებელი კოდები უსასდენო საკომუნიკაციო სისტემებისთვის

საინფორმაციო სისტემები

სამაგისტრო ნაშრომი შესრულებულია საინფორმაციო სისტემების მაგისტრის აკადემიური ხარისხის მოსაპოვებლად

ხელმძღვანელი

ლელა მირცხულავა

დოქტორი,
ასოცირებული პროფესორი

თბილისი, 2013

სარჩევი

ანოტაცია 3

შესავალი 4

ფაქტორები რომლებიც ზეგავლენას ახდენენ ბიტური შეცდომების კოეფიციენტზე და სიმულაციის საშუალებები 7

მოდელირების პროცედურა 9

სიმლულაციის შედეგი 12

არხის კოდირება 15

ჭარბი კოდირება 17

წრფივი კოდების დეკოდირების საერთო მეთოდი 18

ციკლური წრფივი კოდები 19

კონვოლუციური კოდირება 20

ვიტერბის ალგორითმი 24

ხელახალი გადაცემის ავტომატური მოთხოვნა 25

კასკადური კოდირება, იტერაციული დეკოდირება 26

დასკვნა 29

გამოყენებული ლიტერატურა 30

ანოტაცია

წინამდებარე ნაშრომში განიხილება ბიტური შეცდომების კოეფიციენტის, BER(bit error rate)-ის სიმულაცია Matlab-ის გამოყენებით. ბიტური შეცდომების კოეფიციენტი, BER, წარმოადგენს საკვანძო პარამეტრს, რომელიც გამოიყენება ისეთი სისტემების შესაფასებლად, რომლებიც ციფრულ მონაცემებს გადასცემენ ერთი ადგილიდან მეორეში. მათში შედის როგორც რადიო გადამცემი, ასევე ოპტიკურ-ბოჭკოვანი მონაცემთა სიტემები, Ethernet და სხვა სისტემები, რომლებიც ამა თუ იმ სახით მონაცემებს გადასცემს ქსელში სადაც ხმაურს, ინტერფერენციას, ფაზის რხევას შეუძლია ციფრულის სიგნალის ხარისხი შეამციროს. Matlab წარმოადგენს იდეალურ ინსტრუმენტს ციფრული კომუნიკაციების სისტემის მოდელირებისთვის. ერთ-ერთ ყველაზე ხშირად მოდელირებულ ამოცანას წარმოადგენს მოდემის BER-ის გამოთვლა.

Annotation

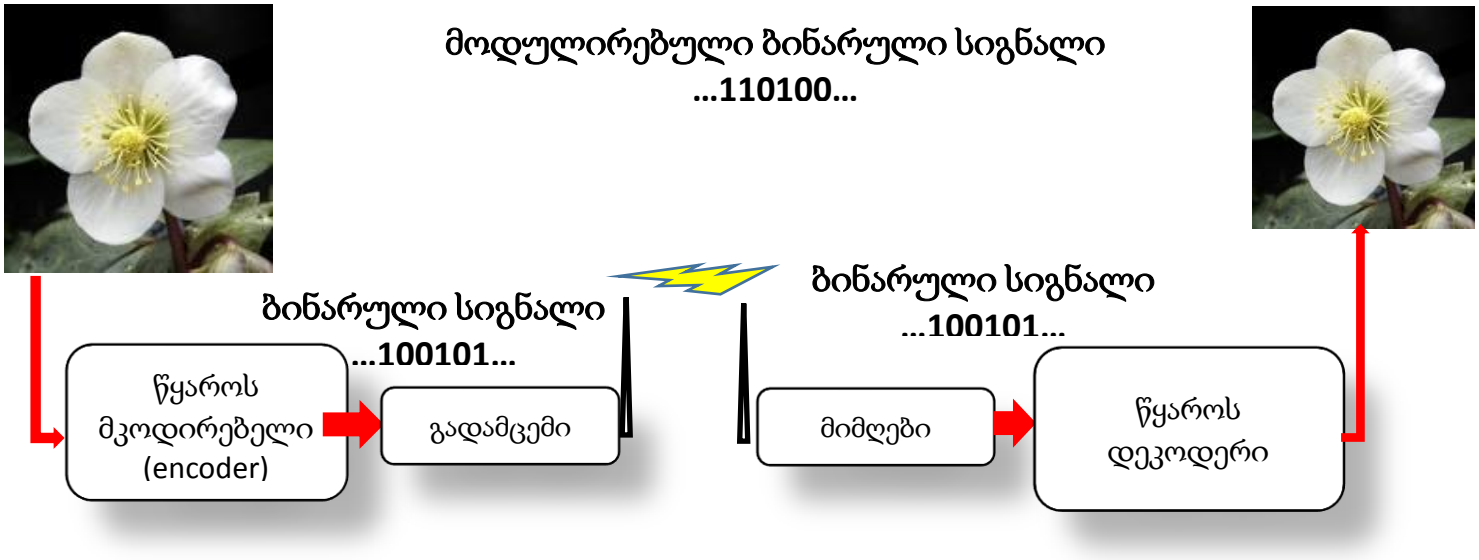
This paper introduce the Bit error rate, (BER) simulation using Mat lab. Bit error rate, (BER) is a key parameter that is used in assessing systems that transmit digital data from one location to another. Systems for which bit error rate, is applicable include radio data links as well as fiber optic data systems, Ethernet, or any system that transmits data over a network of some form where noise, interference, and phase jitter may cause quality degradation of the digital signal. Mat lab is an ideal tool for simulating digital communications systems, thanks to its easy scripting language and excellent data visualization capabilities. One of the most frequent simulation tasks in the field of digital communications is bit-error-rate testing of modems. The bit-error-rate performance of a receiver is a figure of merit that allows different designs to be compared in a fair manner. Performing bit-error-rate testing with Mat lab is very simple, but does require some prerequisite knowledge.

შესავალი

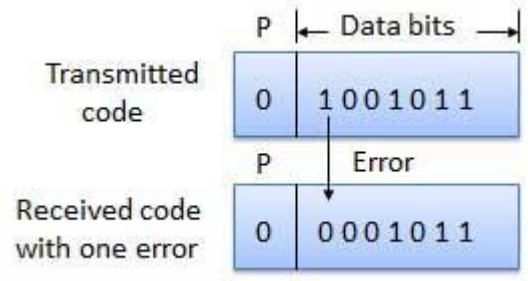
ინტერნეტის შემოსვლიდან ჩვენს ცხოვრებაში ახალი ერა იწყება. სადენების ქსელი, რომლებიც მსოფლიოს მოიცავს უფრო და უფრო სქელი ხდება. თავდაპირველად ეს იყო ტელეგრაფები, ტელეფონებისა და მოდემების წვრილი სადენები, შემდეგ მომსხვილოკაბელიები საკაბელო ტელევიზიისთვისა და ქსელისთვის, შემდეგ ოპტიკურ ბოჭკოვანი წვრილი კაბელები და ბრონირებული მონსტრები ატლანტიკური ოკეანის პსკერზე. საინტერესე გრადიაციაა - პატარა დაწვრილებიდან, დიდ და გრძელებზე და შემდეგ ისევ უჩინარ კავშირამდე. მაგრამ ეს არ არის პროგრესის დასასრული. უსადენო საკომუნიკაციო სისტემების განვითრება შესაძლებლობას გვაძლევს საერთოდ უარი ვთქვათ სადენიან ტექნოლოგიებზე.

რომელი უსადენო კომუნიკაციის სისტემები ვიცით? პირველი რაც აზრად მოგვდის ეს არის რადიო, რომლის დროც დასასრულისკენ მიდის. მობილური კავშირი, რომლის განვითარება 1G-დან დაიწყო და 3G-მდე ავიდა. და რა თქმა უნდა WI-FI ინტერნეტი. ისინი უკვე ფართოდ გამოიყენება ჩვენს ცხოვრებაში. სასჯელ აღსრულება, სასწრაფო სერვსები, ტრანსპორტირება, სამხედრო დანიშნულება.

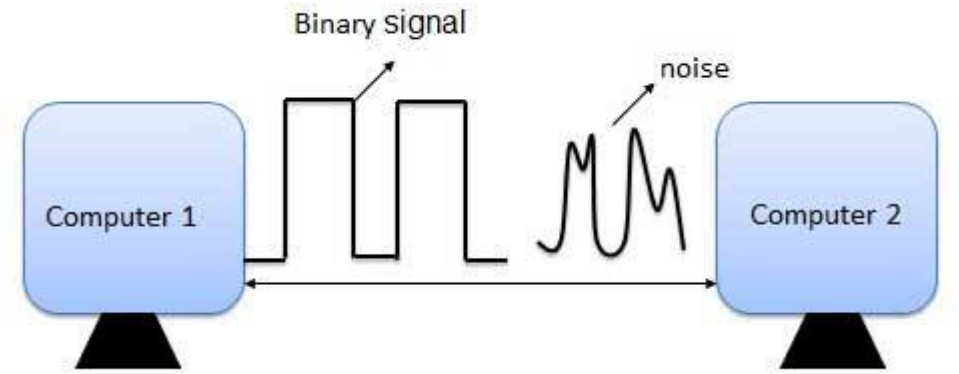
უსადენო კომუნიკაციების გამოყენებისას ხდება სხვადასხვა ტიპის მონაცემების გადაცემა. ისეთების როგორებიცაა ტექსტური შეტყობინებები, ფაილური მონაცემები, სურათი/ვიდეო, საუბარი/მუსიკა. ამ დროს წყარო მკოდირებელი(encoder) ახდენს მონაცემის ბინარულ სიგნალში გარდაქმნას, რათა გადამცემმა გადასცეს მიმღებს. შემდეგ დეკოდერი ახდენს მონაცემის აღდგენას ბინარული სიგნალიდან.



საუბედუროდ, მონაცემთა გადაცემის დროს შესაძლებელია მოხდეს შეცდომები, რომლის მიზეზებიცაა ხმაური, სხვა საკომუნიკაციო მოწყობილობების ინტერფერენცია, სიგნალის მრავალმიმართულებიანი გავრცელება და სხვა. ამ დროს წარმოიქმნება შეცდომები ბინარულ სიგნალში. მაგალითად თუ გასასაცემი მონაცემთა ბიტებია 1 0 0 1 1 0 1 0 1 0 0 0 1 0 1 1 1 0 1 0, შეცდომების შემდეგ მიმღები მიიღებს 1 0 1 1 0 1 0 0 1 . . . შედეგად დეკოდერი ვერ მოახერხებს კორექტულად აღადგინოს გაგზავნილი მონაცემი.



შეცდომების სიმრავლის დროს ეჭვქვეშ დგება უსადენო კომუნიკაციების საიმედოობა. ამ დროს ძნელია აწარმოო აზრიანი საუბარი ტელეფონით, დაზიანებული გამოსახულება და მუსიკა შეამცირებს მომხმარებელთა აუდიტორიას, უსადენო ინტერნეტი ბევრი შეცდომით აძნელებს მის მოხმარებას, ზიანდება ელ. წერილები, შეუძლებელი ხდება საკრედიტო ბარათებით ვაჭრობა და ა.შ. ჩვენი კვლევის მიზანია გავზარდოთ უსადენო კომუნიკაციების საიმედოობა შეცდომების შემცირების გზით. ამისთვის ჩვენ გამოვიყენებთ დაპროგრამების გარემოს Matlab-ს, სიმულაციური სისტემის ასაგებად და მისი ტესტირებისთვის.



როგორც დასახელებიდან ჩანს BER(bit error rate) განისაზღვრება, როგორც გადამცემ სისტემაში შეცდომების წარმოქმნის კოეფიციენტი. ეს პირდაპირ შეიძლება ითარგმნოს, როგორც შეცდომების რაოდენობა, რომლებიც წარმოიქმნება სიტყვაში ბიტების წინასწარ ცნობილი რაოდენობით და აისახება ფორმულით:

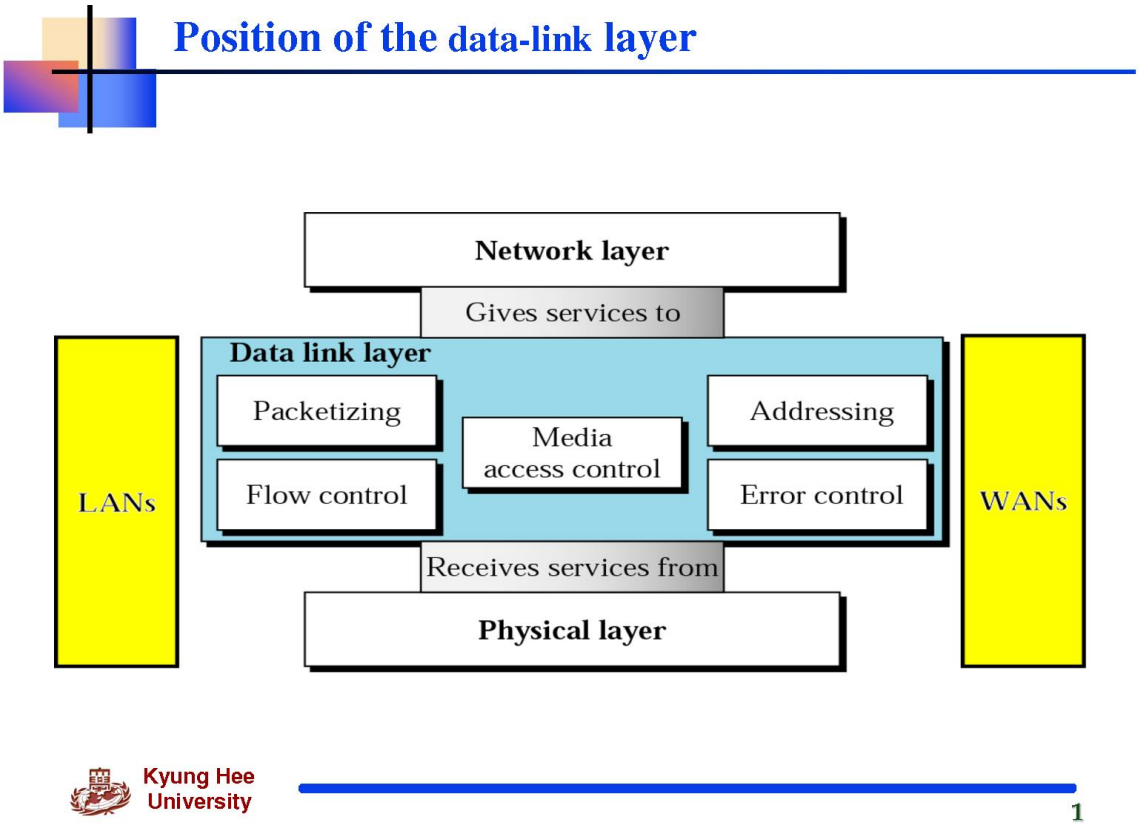
BER = შეცდომების რაოდენობა / გაგზავნილი ბიტების საერთო რაოდენობა

თუ გარემო გადამცემსა და მიმღებს შორის არის კარგი და სიგნალი-ხმაურის შეფარდება მაღალია, მაშინ BER იქნება ძალიან მცირე და სისტემაზე შესამჩნევ გავლენას ვერ მოახდენს. თუმცა თუ ხმაური აღმოჩენილი იქნა, მაშინ არის შანსი BER-ის განხილვა დაგვჭირდეს. მიუხედავად იმისა რომ არსებობს განსხავება სისტემების მუშაობაში და იმაში თუ როგორ მოქმედებს მათზე BER, მისი ძირითადი განსაზღვრება იგივე რჩება. როდესაც მონაცემები გადაეცემა მონაცემთა გადამცემ არხში, შესაძლებელია წარმოიქმნეს შეცდომები სიტემაში. ამ დროს სისტემის მთლიანობა შესაძებელია დაირღვეს. შედეგად აუცილიბელია შევაფასოთ სისტემის წარმადობა და BER ამისთვის იდეალურ გზას უზრუნველყოფს. შეფასების სხვა ფორმებთან შედარებით, BER აფასებს სისტემის სრულ

წარმადობას გადამცემის, მიმღებისა და მათ შორის გარემოს ჩათვლით. ამგვარად BER საშუალებას გვაძლევს გავტესტოთ სისტემის ფაქტიური წარმადობა, იმის მაგივრად რომ ცალკეული კომპონენტები გაგვეტესტა და იმედი გვქონოდა, რომ ერთად დამაკმაყოფილებლად იმუშავებდნენ. მონაცემთა გადამცემი არხის და შესაბამისად BER-ის გაუარესების ძირითადი მიზეზებია ხმაური და ცვლილებები გავრცელების გზაზე(იქ სადაც რადიო სიგნალები გამოიყენება). არხის მახასიათებლების ანალიზი ჩვეულებრივ სტატისტიკური ანალიზის მეთოდების გამოყენებით იწარმოება. ოპტიკურ-ბოჭკოვანი სისტემებში ბიტური შეცდომები წარმოიქმნება არხის შექმნისთვის საჭირო კომპონენტების ნაკლებობით. მათში შედის ოპტიკური დრაივერი, მიმღები, გადამცემი და თავად ბოჭკოვანი კაბელი. ბიტური შეცდომები შეიძლება ასევე წარმოიქმნას ოპტიკური დისპერსიის და სიგნალის ჩაქრობის შედეგად. ამასთან ერთად ხმაურ შეიძლება წარმოიქმნას თავად ოპტიკურ მიმღებშიც. ჩვეულებრივ ისინი შეიძლება იყვნენ ფოტოდიოდები ან გამაძლერებლები, რომლებმაც უნდა უპასუხონ ძალიან მცირე ცვლილებებს და შედეგად იქმნება ხმაური. კიდევ ერთი ფაქტორი ბიტური შეცდომებისთვის - ფაზის ნებისმეიერი რხევა, რომელიც არის სისტემაში, რადგან მას შეუძლია შეცვალოს თავდაპირველი მონაცემები. შესაბამისობა სიგნალი-ხმაური და რიცხვები Eb/No - პარამეტრებია რომლებიც უფრო დაკავშირებულია რადიოკავშირის და რადიო კომუკიაციების სისტემებთან. ამ თვალსაზრისით BER შეიძლება ასევე აღიწეროს, როგორც შედცომის წარმოქმნის ალბათობა ან უბრალოდ POE(probability of error). მის განსასაზღვრად სამი სხვა პარამეტრი გამოიყენება. ისინი არიან შეცდომების ფუნქცია(erf), სიმძლავრე ერთ ბიტში(Eb) და ხმაურის სიმძლავრე გამტარუნარიანობაში 1Hz-ზე(No). უნდა ავღნიშნოთ, რომ ყოველი განსხვავებული ტიპის მოდულაციას გააჩნია თავისი საკუთარი მნიშვნელობა შეცდომების ფუნქციისთვის. ამის მიზეზია ის, რომ მოდულაციის ყოველი ტიპი სხვადასხვანაირად იქცევა ხმაურის არსებობის დროს. კერძოდ, უმაღლესი რიგის მოდულაციი სქემები(მაგალითად 64QAM და ა.შ), რომლებსაც შეუძლიათ ტვირთონ მონაცემთა გადაცემის მაღალი სიჩქარეები, არც ისე მდგრადებია ხმაურის არსებობის შემთხვევაში. დაბალი რიგი მოულაციის ფორმატები(BPSK, QPSK და ა.შ) გვთავაზობენ მონაცემთა გადაცემის უფრო დაბალ სიჩქარეს, მაგრამ უფრო მდგრადებია. სიმძლავრე ერთ ბიტზე(Eb), შეიძლება განისაზღვროს, როგორც გამტარუნარიანობა გაყოფილი მონაცემთა გადაცემის სიჩქარეზე და გაიზომება ჯოულებში. No არის სიმძლავრე ერთ ჰერცზე და განზომილებაა ჯოული წამში გაყოფილი წამებზე. Eb/No თანაფარდობის განხილვისას ყველა განზომილება ბათილდება. მნიშვნელოვანია ავღნიშნოთ, რომ POE პროპორციულია Eb/No-ის და წარმოადგენს სიგნალი-ხმაურის შესაბამისობის ფორმას.

ფაქტორები რომლებიც ზეგავლენას ახდენენ ბიტური შეცდომების კოეფიციენტზე და სიმულაციის საშუალებები

Eb/No გამოყენებიდან შესამჩნევია, რომ ბიტური შეცდომების კოეფიციენტზე(BER) ბევრი ფაქტორი ახდენს ზეგავლენას. კონტროლირებადი ცვლადების მანიპულირებით შესაძლებელია სისტემის ოპტიმიზირება საჭირო წარმადობის უზრუნველსაყოფად. ჩვეულებრივ ეს მონაცემთა გადამცემი სისტემის პროექტირების ეტაპზე ხდება ასე, რომ წარმადობის პარამეტრები შეიძება ვარეგულიროთ საწყისი კონცეპტების განსაზღვრის დროს. სისტემაში არსებული ინტერფერენციის დონეები ჩვეულებრივ განისაზღვრება გარე ფაქტორებით და შეუძლებელია შეიცვალოს სისტემის პროექტირებით. თუმცა შესაძლებელია სისტემის გამტარუნარიანობის დაყენება. გამტარუნარიანობის შემცირებით შესაძლებელია ინტერფერენციის დონის შემცირება. ამის გარდა შესაძლებელია სისტემის სიმძლავრე გავზარდოთ ისე, რომ სიმძლავრე ერთ ბიტზე(Eb) გაიზარდოს. ეს ბალანსირებული უნდა იყოს ისეთ ფაქტპრებთან როგორებიცაა სხვა მომხმარებლების ინტერფერენცია, საერთო ენერგიის მოხმარება და ა.შ. დაბალი რიგის მოდულაციის სქემების გამოყენება შესაძლებელია, მაგრამ გამტარუნარიანობის ხარჯზე. საჭიროა დავაბალანსოთ ყველა ფაქტორი, რათა მივიღოთ დამაკმაყოფილებელი BER. ჩვეულებრივ ყველა მოთხოვნის დაკმაყოფილება შეუძლებელია და საჭიროა რაღაც კომპრომისებზე წავიდეთ.



MATLAB წარმოადგენს გამოთვლით გარემოს და მეოთხე თაობის დაპროგრამების ენას. Math Works კომპანიის შემუშავებული, MATLAB საშუალებას გვაძლევს ვაწარმოოთ მატრიცული მანილუპაციები, ფუნქციების და მონაცემების გრაფიკული ასახვა, ალგორითმების რეალიზაცია, სამომხმარებლო ინტერფეისების შექმნა და ამ ინტერფეისებით ურთიერთქმედება პროგრამებთან დაწერლი სხვა ენებზე C, C++, JAVA და სხვა. მიუხედავად იმისა რომ მათლაბი პირველ რიგში გამოიყენება რიცხვითი გამოთვლებისთვის, დამატებითი ინსტრუმენტების პანელი იყენებს MuPAD სიმბოლურ მექანიზმს რაც საშუალებას გვაძლევს გამოვიყენოთ სიმბოლური გამოთვლითი შესაძლებლობები. ერთე-ერთ ყველაზე ხშირ მოდელირების ამოცანას ციფრული კომუნიკაციების სფეროში წარმოადგენს მოდემის BER-ის გატესტვა. ბიტური შეცდომების კოეფიციენტის მწარმოებლობა ხარისხის მაჩვენებელია, რომელიც სხვადასხვა პროექტების სამართლიანი შედარების საშუალებას გვაძლებს. მათლაბში ჩვენ ერთჯერად-შეუწყვეტელ სიგნალებს წარმოვადგენთ, როგორც რიცხვების მიმდევრობას რომლებიც შენახულია ვექტორში ან მასივში. სანამ ჩვენ შევამოწმებდით BER-ს, ჩვენ უნდა გავიგოთ ამ რიცხვთა მიმდევრობების მნიშვნელობა. ჩვენ უნდა ვიცოდეთ სიგნალის რა ასპექტს წარმოადგენს თითოეული მათგანი. ჩვენ ასევე უნდა ვიცოდეთ დროის ინტერვალი თანმიმდევრულ ნიმუშებს შორის. კომუნიკაციური მოდელირებისთვის, ნიმუშთა რიცხვითი მნიშვნელობები წარმოადგენს ერთჯერად-უწყვეტ სიგნალის ამპლიტუდებს დროის გარკვეულ მომენტში. ჩვენ ვვარაუდობთ, რომ ეს ამპლიტუდა ძაბვის გაზომილებაა, თუნცა ის ასევე შეიძლება იყოს დენის გაზომილება. დრო თანმიმდევრულ ნიმუშებს შორის იყოს Ts. ეს შეგვატყობინებს რამდენად ხშირად უწყვეტი სიგნალი იქნა არჩეული. იმის მაგივრად რომ განვსაზღვროთ Ts, ჩვენ ჩვეულებრივ ვპოულობთ დისკრეტიზაციის სიხშირეს, fs, რომელიც არის Ts-ის ინვერსია. მოხერხებულობისთვის ჩვენ ყოველთვის შევუსაბამებთ სანიმუშო მნიშვნელობას 1.0 ერთ ვოლტიან დაძაბულობას. ამის გარდა, წინააღმედოგაბა ყოველტვის გვექნება 1 ომი. ეს საშუალებას მოგვცემს საერთო ჯამში უგულებელვყოთ წინააღმდეგობა. ჩვენ მოდელირებისთვის უწყვეტ სიგნალს წარმოვადგენთ როგორც ნიმუშთა მასივს, რომელთა რიცხვითი მნიშვნელობები არის ვოლტების ერთეულებში ერთ ომიანი წინააღმდეგობით. ჩვეულებრივ დისკრეტიზაციის სიხშირე შეადგენს 8 KHz-ს, მაგრამ დისკრეტიზაციის სხვა სიხშირეებიც ფართოდ გამოიყენება, ამგვარად ის ყოველთვის უნდა განისაზღვროს. წარმოვიდგინოთ რომ ჩვენ გვაქვს სიგნალი x(n), სადაც n არის ნიმუშის ინდექსი. სიგნალის მყისიერ სიმძლავრეს განვსაზღვავთ როგორც:

$Pins \equiv x^2(n)$.

სხვა სიტყვებით ნიმუშის მყისიერი სიმძლავრე ამ ნიმუშის მნიშვნელობის კვადრატია. რახან ნიმუშის ერთეულია ვოლტი, სიმძლავრის ერთეული იქნება ვატი. უფრო სასარგებლოა საშუალო სიმძლავრე, რომელიც ყოველი ნიმუშის მყისიერი სიმძლავეების საშუალოს წარდმოადგენს. N ნიმუშის x(n) სიგნალისთვის ჩვენ გვაქვს:

$$Pave \equiv \frac{1}{N}\sum_{n=1}^{N} x^2(n) \quad (1)$$

ერთე-ერთი გზა ამ ფორმულის მათლაბში გამოთვლის არის:

pav= sum(x.^2)/length(x).

თუ ჩვენი სიგნალის საშუალო მნიშვნელობა ნულია, ჩვენ შეგვიძლია სიგნალის საშუალო სიმძლავრე გამოვთვალოთ სხვაობის აღებით. ეს მუშაობს იმიტომ, რომ:

$$\sigma(x) \equiv E[x^2] - (E[x])^2,$$

თუ საშუალო არის ნული, სხვაობა არის კვადრატების საშუალო, ზუსტად ისევე როგორც საშუალო სიმძლავრე. სიგნალის საშუალო სიმძლავრის გამოთვლა სხვაობის მეშვეობით ფრთხილად უნდა გამოვიყენოთ. ეს მხოლოდ მაშინ მოქმედებს, როცა სიგნალის საშუალო მნიშვნელობა ნულია. თუ საშუალო მნიშვნელობა მისგან განსხვავდება, უნდა გამოვიყენოთ (1) ფორმულა, რომელიც ყოველთვის მოქმედებს იმისდა მიუხედავად საშუალო მნიშვნელობა ნულია თუ არა. ამის გარდა სიგნალის ხანგრძლივობა წარმოადგენს ნიმუშის სიგძეს გაყოფილს დისკრეტიზაციის სიხშირეზე, ამგვარად:

$$E_{tot} = Pave * t = \frac{1}{N}\sum_{n=1}^{N} x^2(n) * \frac{N}{fs} = \frac{1}{fs}\sum_{n=1}^{N} x^2(n)$$

მათლაბის ბრძანება საერთო სიმძლავრის საპოვნელად, et, სადაც სიგნალია x, და დისკრეტიზაციის სიხშირე f_s:

et= sum(x.^2)/ f_s.

მოდელირების პროცედურა

BER-ის ტესტირებისთვიოს საჭიროა გადამცემი, მიმღები და არხი. ჩვენ დავიწყებთ შემთვევითი ბიტების მიმდევრობის გენერაციას, რომელსაც გამოვიყენებთ როგორც შემავალ მონაცემებს გადამცემში. გადამცემი გარდაქმნის ამ ბიტებს ციფრული სიგნალების რაღაც ფორმაში, რომელსაც ჩვენ გავაგზავნით სიმულაციურ არხში. ჩვეულებრივ BER-ის წარმადობა გამოისახება ორგანზომილებიანი გრაფიკით. ორდინატა არის Eb/N0: ენერგია ერთ ბიტზე გაყოფილი ხმაურის ერთმხრივ სპექტრალურ სიმჭიდროვეზე გამოსახული დეციბელებში(dB). აბსცისა ბიტური შეცდომების კოეფიციენტი, განზომილების გარეშე, ჩვეულებრივ წარმოდგება ათის ხარისხებში. BER-ის SNR(normalized signal-to-noise ratio)-ზე გრაფიკის ასაგებად ჩვენ ავაგებთ წერტილების მიმდევრობას. ყოველი ეს წერტილი მოითხოვს რათა ჩვენ მოვახდინოთ მოდელირება SNR-ის გარკვეული მნიშვნელობისას. იმისთვის რომ მივიღოთ ბიტური შეცდომების კოეფიციენტი განსაზღვრულ SNR-ზე, ჩვენ ქვემოთ მოყვანილ პროცედურას ვასრულებთ:

A. **გადამცემის გაშვება:** თავდაპირველად ჩვენ უნდა გამოვიყენოთ გადამცემი ციფრულ ფორმაში მოდულირებული სიგნალის შესაქმნელად პსევდოშემთხვევითი ბიტების მიმდევრობიდან.
B. **SNR-ის განსაზღვრა:** SNR ანუ E_b/N_0 ჩვეულებრივ გამოისახება დეციბელებში, მაგრამ ჩვენ უნდა გარდავქმნათ დეციბელები ჩვეულებრივ კოეფიციენტში, სანამ

შევძლებთ მის გამოყენებას. თუ ჩვენ დავაყენებთ SNR-ს m dB-ს, მაშინ E_b/N_0= 10m/10. მათლაბის გამოყენებით:

ebn0= 10^(snrdb/10)

ჩვენ ვპოულობთ კოეფიციენტს, ebn0, დეციბელებიან SNR-იდან,snrdb. შევნიშნოთ რომ E_b/N_0 -ს განზომილება არ გააჩნია.

C. **განვსაზღვროთ E_b**: სიმძლავრე ერთ ბიტზე არის სიგნალის საერთო სიმძლავრე გაყოფოლი სიგნალში ბიტების რაოდენობაზე. მისი წარმოდგენა შემდეგნაირად შეიძლება:

$$E_b = \frac{1}{N * f_{bit}} \sum_{n=1}^{N} x^2(n)$$

სადაც N არის ნიმუშების საერთო რაოდენობა სიგნალში, და f_{bit} გადაცემის სიჩქარე წამში. მათლაბის გამოყენებით:

eb = sum(x.^2)/(length(x)*fb).

E_b -ის განზომილება იქნება ჯოული.

D. **N_0-ის გამოთვლა:** SNR-ის და E_b-ის გამოთვლის შემდეგ დროა გამოვთვალოთ N_0. ერთადერთი რაც უნდა გავაკეთოთ ეს არის გავყოთ E_b SNR-ზე იმ შემთხვევაში თუ SNR-ის დეციბელები გადავიყვანეთ კოეფიციენტში. მათლაბის გამოყენებით:

n0 = eb/ebn0.

განზომილება გააჩნია ვატი ჰერცზე.

E. **σ_n-ის გამოთვლა:** N_0 გვეუბნება ჩვენ თუ რა სიმძლავრის ხმაური არის წარმოდგენილი 1.0 Hz გამტარ ზოლში. იმისთვის, რომ გავიგოთ ხმაურის დისპერსია ან საშუალო სიმძლავრე ჩვენ უნდა ვიცოდეთ ხმაურიანი გამტარუნარიანობა(the noise bandwidth). რეალური სიგნალისთვის, x(n), ხმაურიანი გამტარუნარიანობა იქნება დისკრეტიზაციის სიხშირის ნახევარი.

$$\sigma_n = \frac{N0 * fs}{2}$$

სადაც σ_n არის ხმაურის დისპერსია W-ში. გადავიყვანოთ მათლაბში:

pn= n0*fs/2.

განზომილება იქნება ვატში.

F. **ხმაურის გენერაცია:** მიუხედავად იმისა, რომ მათლაბს გააჩნია საშუალებები დამატებითი გაუსური თეთრი ხმაური, ჩვენ გამოვიყენებთ სტანდარტულ ჩაშენებულ ფუნქციას AWGN-ის გენერაციისთვის. რადგანაც ხმაურს გააჩნია ნულოვანი საშუალო მნიშვნელობა მისი სიმძლავრე და დისპერსია იქნება ერთიდაიგივე. ჩვენ უნდა დავაგენერიროთ ხმაურის ვექტორი, რომლის სიგრძე ჩვენი ორიგინალური ვექტორის სიგრძის ტოლი იქნება და ამ ვექტორის დისპერსია უნდა იყოს σ_n W. მათლაბის ფუნქცია randn აგენერირებს შემთხვევით რიცხვებს საშუალო მნიშვნელობით ნულით და დისპერსიით ერთი. ჩვენ უნდა მოვახდინოთ გამომავალის მასშტაბირება იმგვარად რომ შედეგს ჰქონდეს საჭირო დისპერსია σ_n. ამისთვის უბრალოდ გავამრავლოთ randn-ის შედეგი p¾n-ზე.

n = sqrt(pn)*randn(1,length(x));

როგორც სიგნალების ვექტორს აქაც განზიმილებაა ვოლტი.

G. **ხმაურის დამატება:** ჩვენ ვქმნით ხმაურიან სიგნალს ხმაურის ვექტორის დამატებით სიგნალის ვექტორს. თუ ჩვენ ვახდენთ ფიქსირებული წერტილის სიმულაციას, მაშინ ჩვენ გვიწევს შედეგის მასშტაბირება. შხვა შემთხვევაშ ჩვენ უბრალოდ ვამატებთ ხმაურის ვექტორს, n, სიგნალის ვექტორს, x, რათა მივიღოტ ხმაურიანი ვექტორი, y: y = x+n

H. **მიმღების გაშვება და წანაცვლების განსაზღვრა:** მას შემდეგ რაც შევქმენით ხამურიანი სიგნალი ჩვენ ვიყენებთ მიმღებს მისი დემოდულაციისთვის. მიმღების აწარმოებს დემოდულირებული ბიტების მიმდევრობას, რომლებიც უნდა შევადაროთ გადაცემულ ბიტებს, რომ გავარკვიოთ რამდენი დემოდულირებული ბიტია შეცდომით. ფილტრაციის და სხვა შეფერხებების გამომწვევი ოპერაციების გამო, ტიპიური უმეტესი მიმღებებისთვის შეიძლება წანაცვლება იყოს მიღებულ ბიტებსა და გაგზავნი ბიტებს შორის. მანამ სანამ ჩვენ შევადარებთ ბიტების ორ მიმდევრობას შეცდომების აღმოსაჩენად, ჯერ უნდა განისაზღვროს ეს წანაცვლება. ერთ-ერთი გზა ამის გასაკეთებლად არის ორი მიმდევრობის კორელაცია და შემდეგ ვეძიოთ კორელაციის უმაღლესი წერტილი. დავუშვათ, რომ ჩვენი გადაცემული ბიტები შენახულია tx ვექტორში, ხოლო მიღებული ბიტები rx ვექტორში. მიღებული ვექტორი უნდა შეიცავდეს უფრო მეტ ბიტებს, ვიდრე გადაცემული ვექტორი, რადგან მიმღები აწარმოებს უაზრო შედეგებს, სანამ ფილტრები ივსება და იცლება. თუ გადაცემული ბიტების ვექტორის სიგრძე არის ltx და მიღებული ვექტორის სიგრძე lrx, შესაძლო წანაცვლების დიაპაზონია ნულსა და lrx–ltx–1 შორის. ჩვენ შეგვიძლია ვიპოვოთ წანაცვლება ნაწილობრივი ჯვარედინი კორელაციით ამ ორი ვექტორს შორის. მათლაბის გამოყენებით ჩვენ შეგვიძლია შევქმნათ ეს ნაწილობრივი ჯვარედინი კორელაცია, cor, შემდეგი ციკლის მეშვეობით:

for lag= 1 : length(rx)–length(tx)–1, cor(lag)= tx*rx(lag : length(tx)–1+lag)’; end.

მიღებული ვექტორი, წარმოადგენს ნაწილობრივ ჯვარედინ კორელაციასგადაცემული და მიღებული ბიტებისა შესაძლო ლაგების დიაპაზონით: 0 : *lrx –lt x* –1. ჩვენ უნდა ვიპოვოთ cor-ის მაქსიმალური მნიშვნელობის ადგილმდებარეობა, რადგანაც ეს გვეტყვის წანაცვლებას ბიტურ ვექტორებში. მათლაბი გამოყენებით მივიღებთ:

off= find(cor== max(cor))–1.

I. **შეცდომების ვექტორის შექმნა:** როგორც კი გავიგებთ წანაცვლებას გადაცემულ და მიღებულ ბიტურ ვექტორებს შორის, ჩვენ მზად ვართ გამოვთვალოთ ბიტური შეცდომები. ბიტური მნიშვნელობისთვის ნული და ერთი, მარტივი განსხვავება გვაჩვენებს ბიტურ შეცდომას. ყველგან სადაც იქნება შეცდომა ბიტებს შორის განსხვავება იქნება ±1 და ყველგან სადაც შეცდომა არ არის სხვაობა იქნებ ნული. მათლაბის გამოყენებით ჩვენ გამოვთვლით შეცდომების ვექტორს, err, გადაცემულ ვეტორს, tx, და მიღებულ ვექტორ, rx, შორის რომელსაც გააჩნია წანაცვლება off:

err = tx–rx(off+1 : length(tx)+off);.

J. **დავითვალოთ ბიტების შეცდომები:** შეცდომების ვექტორი, err, შეიზცავს არანულოვან ელემენტებს იმ ადგილებში სადაც ბიტური შეცდომები მოხდა. ჩვენ

უნდა დავითვალოთ არანულოვანი ელემენტების რიცხვი, რადგან ესაა მცდარი ბიტების საერთო რაოდენობა ამ სიმულაციაში. მათლაბის გამოყენებით:

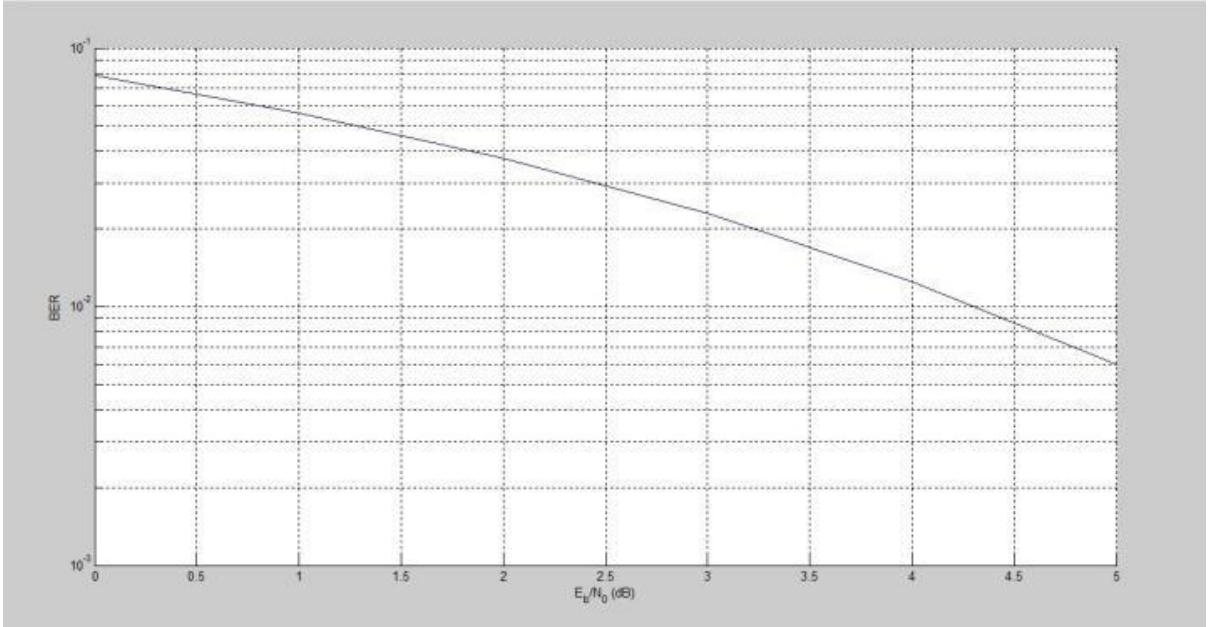
te= sum(abs(err)).

K. **ბიტური შეცდომების კოეფიციენტის გამოთვლა:** ყოველთვის როცა ჩვენ ვუშვებთ BER-ის სიმულაციას ჩვენ გადავცემთ და ვიღებთ ბიტების მიუდმივ რაოდენობას. ჩვენ ვარკვევთ რამდენია მიღებულ ბიტებს შორის ნცდარი და შემდეგ გამოვითვლით BER-ს როგორც მცდარი ბიტების რაოდენობა გაყოფილი ბიტების საერთო რაოდენობაზე გადაცემულ სიგნალში. მათლაბში: ber= te/length(tx), სადაც te მცდარი ბიტების საერთო რაოდენობაა, ხოლო tx კი გადაცემული ბიტების ვექტორი.

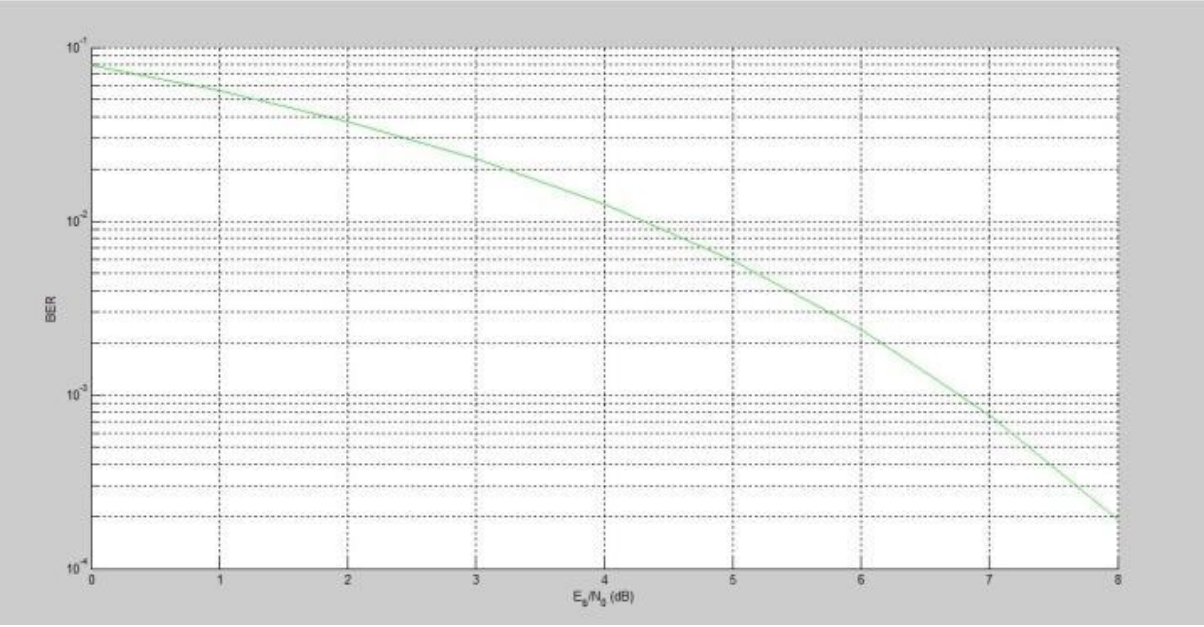
სიმლულაციის შედეგი

BER-ის მოდელირების შესრულება გრძელი პროცესი შეიძლება იყოს. ჩვენ უნდა მოვახდინოთ ცალკეული სიმულაცია საინტერესო SNR-თვის. ჩვენ ასევე უნდა დავრწმუნდეთ, რომ მიჭებული შედეგები სტატისტიკურად მნიშვნელოვანია. როდესაც BER არის მაღალი, მაშინ მრავალი ბიტი დაზიანებულია. BER-ის ყველაზე ცუდი მნიშვნელობაა 50 პროცენტი, რომლის შემდეგაც მოდემი პრაქტიკულად უსარგებლოა. კომუნიკაციური სისტემების უმრავლესობა BER-ის გაცილებით ნაკლებ მნიშვნელობებს ითხოვს. ერთი პროცენტიც კი ზოგჯერ ძალიან მაღალ მაჩვენებლად ითვლება. ჩვენთვის საჭიროა გრაფიკულად გამოვსახოთ BER-ის მრუდი, როგორც SNR-ის ფუნქცია და ჩავრთოთ საკმარისი წერტილები BER-ის ფართო დიაპაზონის დასაფარად. მაღალი SNR-ის შემთხვევაში ეს გართულებული შეიძლება იყოს, რაგდან ბიტური შეცდომების კოეფიციენტი ძალიან მცირე ხდება. მაგალითად BER-ის მნიშვნელობაა 10-6, ნიშნავს მხოლოდ ერთ მცდარ ბიტს ერთ მილიონ ბიტში. თუ ჩვენი სატესტო სიგნალი შეიცავს მხოლოდ 1000 ბიტს, მაშინ დიდი ალბათობით შეცდომას ვერ დავინხავთ. იმისთვის რომ ჩვენი მოდელირება იყოს სტატისტიკურად მნიშვნელოვანი იყოს საჭიროა რაღაც რაოდენობის შეცდომების გენერაცია. თუ მოდელირება არ აგენერირებს შეცდომებს ეს იმას არ ნიშნავს, რომ BER ნულია. ეს მხოლოდ იმაზე მიანიშნებს რომ ჩვენ არ გვქონდა საკმარისი ბიტების რაოდენობა გადაცემულ სიგნალში. როგორც გამოცდილება გვიჩვენებს ჩვენ გვჭირდება დაახლოებით 100(ან მეტი) შეცდომა თითოეულ სიმულაციაში, დარწმუნელულები რომ ვიყოთ რომ ჩვენი BER სტატისტიკურად დამაკმაყოფილებელია. მაღალი SNR-ის დროს ეს მოითხოვს ტესტურ სიგნალს, რომელიც შეიცავს მილიონობით, მილიარდობით ბიტს. როგორც კი მივიღებთ საკმარის შედეგებს, საჭიროა ისინი გრაფიკზე ავსახოთ. ჩვენ ვიწყებთ ორივე ღერძისთვის ვექტორების შექმნას. X ღერძი შეიცავს SNR-ის მნიშვნელობებს, ხოლო Y ღერძი კი BER-ს. Y ღერძი გრაფიკზე გამოსახული უნდა იყოს ლოგარითმულ შკალაზე, ხოლო X ღერძი კი წრფივზე. მათლაბში შემდეგნაირად ვიქცევით: semilogy (xx,yy,'o'). სადაც xx SNR-ის ვექტორია, ხოლო

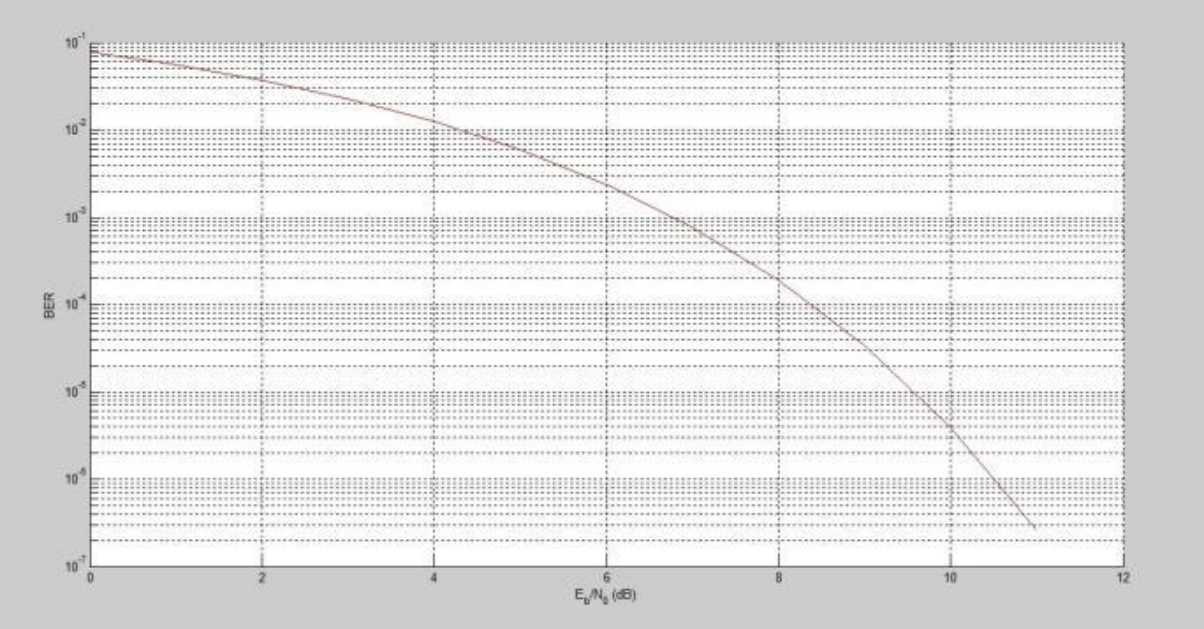
yy კი BER-ის. 1, 2 და 3 სურათები გვიჩვენებს მაგალითს სხვადასხვა Eb/N0-თვის.



სურ. 1



სურ. 2



სურ. 3

მათლაბის კოდი BER-ის სიმულაციისთვის:

N = 10^6 - ბიტების ან სიმბოლოების რიცხვი.

rand('state',100); - rand() ფუნქციის ინიციალიზაცია.

randn('state',200); - randn() ფუნქციის ინიციალიზაცია.

გადამცემისთვის:

ip = rand(1,N)>0.5; - 0,1-ების გენერაცია თანაბარი ალბათობით.

s = 2*ip-1; - BPSK მოდულაცია 0 -> -1; 1 -> 1.

n = 1/sqrt(2)*[randn(1,N) + j*randn(1,N)]; - გაუსის თეთრი ხმაური 0dB დისპერსიით.

Eb_N0_dB = [-3:10];

for ii = 1:length(Eb_N0_dB)

y = s + 10^(-Eb_N0_dB(ii)/20)*n;

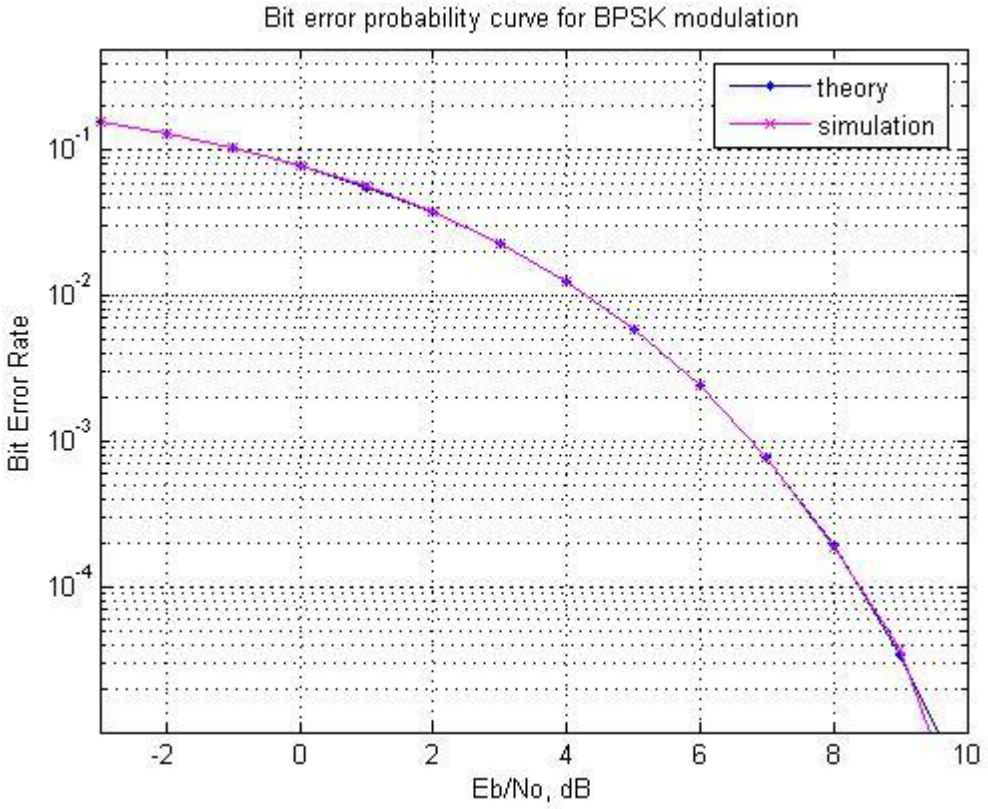
მიმღები:

ipHat = real(y)>0;

nErr(ii) = size(find([ip- ipHat]),2); end

simBer = nErr/N; % simulated ber theoryBer = 0.5*erfc(sqrt(10.^(Eb_N0_dB/10)));

close all figure semilogy(Eb_N0_dB,theoryBer,'b.-'); hold on semilogy(Eb_N0_dB,simBer,'mx-'); axis([-3 10 10^-5 0.5]) grid on legend('theory', 'simulation'); xlabel('Eb/No, dB'); ylabel('Bit Error Rate'); title('Bit error probability curve for BPSK modulation');



არხის კოდირება

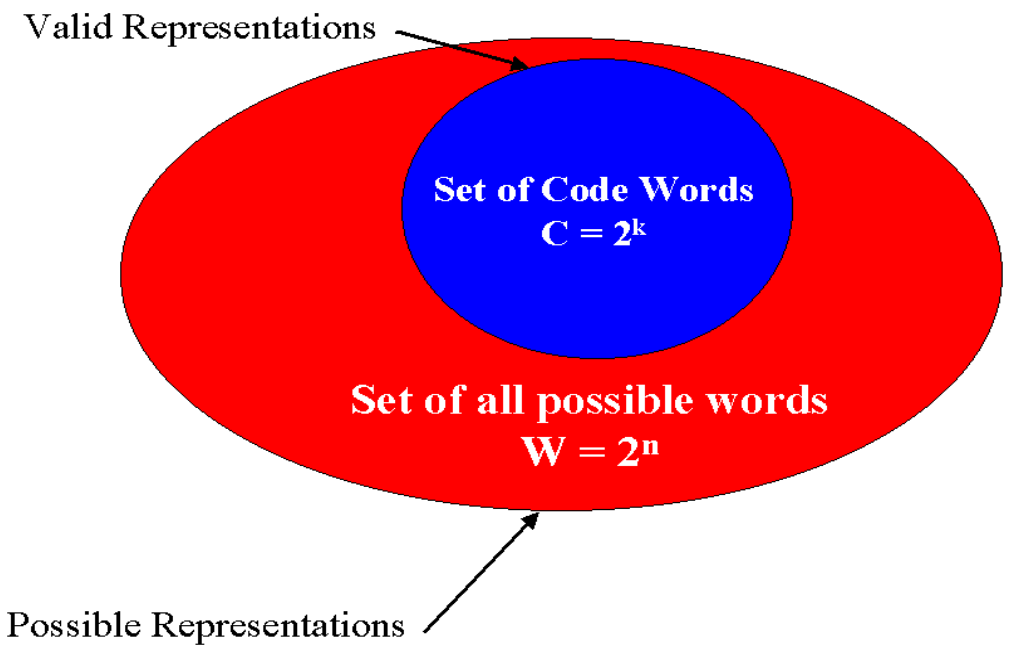
როგორც ვხედავთ BER-ის მაჩვენებელი შეიძლება ძალიან მაღალი იყოს რა ამცირებს უსადენო კომუნიკაციის საიმედოობას, რა შეიძლება გავაკეთოთ მის გასაზრდელად? ჩვენ შევიძლია უარი ვთქვათ უსადენო კომინუკაციებზე ან ვიპოვოთ ხერხი შევამციროთ შეცდომები მიღებულ სიგნალში. შეცდომების აღმოჩენა ეს ქმედებაა მიმართული მონაცემთა მთლიანობის კონტროლზე მისი ჩაწერისას ან გადაცემისას კომუნიკაციის არხში. შეცდომების კორექცია კი ინფორმაციის აღდგენის პროცედურაა მისი წაკითხვისას შენახვის მოწყობილობიდან ან არხიდან. შეცდომბის აღმოსაჩენად იყენებენ შეცდომების აღმომჩენ კოდებს, ხოლო კორექციისთვის კი შეცდომების მაკორექტირებელ კოდებს. უკანასკნელს ეწოდება შეცდომების მაკორექტირებელი კოდირება, ან უბრალოდ არხის კოდირება. შეცდომებთან ბრძოლის რამოდენიმე ხერხი არსებობს:

- შეცდომების აღმოჩენა მონაცემთა ბლოკებში და ხელახალი გადაცემის ავტომატიური მოთხოვნა. ემ მიდგომა გამოიყენება ძირითადად არხის ან ტრანსპორტულ დონეზე.

- შეცდომების აღმოჩენა მონაცემთა ბლოკშეი და დაზიანებული ბლოკების გადაგდება. გამოიყენება მულტიმედიაში სადაც მნიშვნელოვანია გადაცემის დაყოვნება და განმეორებითი გადაცემის დრო არ არის.
- შეცდომების გასწორება, გამოიყენება ფიზიკურ დონეზე.

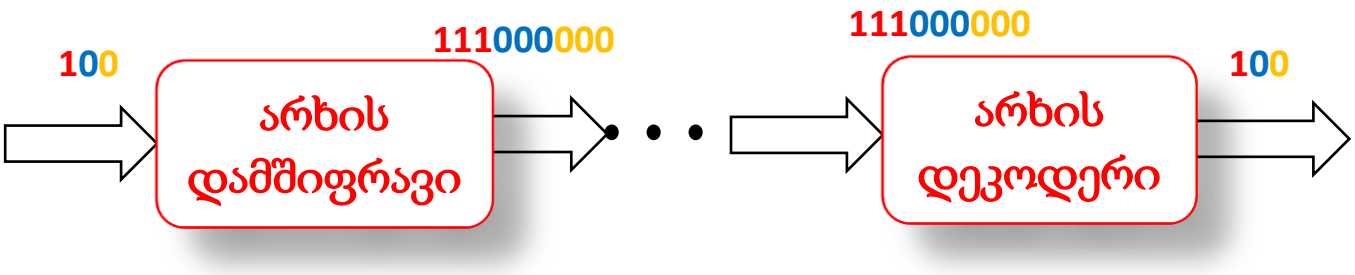
დავუბრუნდეთ მაკორექტირებელ კოდებს. ამ დროს ხდება დამატებითი ბლოკების დამატება გადამცემ და მიმღებ მხარეს(იხ. სურათი). გადამცემ მხარეს ვამატებთ არხის მკოდირებელს, ხოლო მიმღებ მხარეს კი არხის დეკოდერს. არხის მკოდირებელი მართავს საწყის მონაცემებს ისე, რომ შევამციროთ მიღებულ სიგნალში შეცდომების წარმოქმნის ალბათობა. არხის დეკოდერი დაწერილია მკოდირებელთან კოორდინაციაში საწყისი მონაცემების კორექტულად აღდგენისთვის. შეცდომების ალბათობის შემცირება შესაძლებელია ასევე გადაცემის სიჩქარის შემცირების ხარჯზე. მოკერექტირებელი კოდები მჭიდრო კავშირშია შეცდომების აღმომჩენ კოდებთან. პირველისგან განსხვავებით უკანასკნელს შეუძლია მხოლოდ აღმოაჩინოს შეცდომა და არა გაასწოროს ის. სინამდვილეში შეცდომების აღმომჩენი კოდები იმავე კლასს მიეკუთვნება, რაც მაკორექტირებელი კოდები. ფაქტრიურად ნებისმიერი მაკორექტირებელი კოდი აღმომჩენიცაა. მონაცემებთან მუშაობის ხერხების მიხედვით მაკორექტირებელი კოდები შემდეგნაირად იყოფა:

- ბლოკური კოდები, რომლებიც ინფორმაციას ყოფენ მუდმივი სიგრძის ფრაგმენტებად და თითოელ მათგანს ცალ-ცალკე ამუშავებენ.
- კონვოლუციური კოდები, რომლებიც მონაცემებთან მუშაობენ როგორც უწყვეტ ნაკადთან.



ჭარბი კოდირება

განვიხილოთ არხის კოდირების ერთ-ერთი მაგალითი - ჭარბი კოდირება. ჭარბი კოდირება ბლოკური კოდირების მაგალითია. ჭარბი კოდირების დროს n-სიჭარბის კოდირებაში, ყოველი ბიტი კოდირდება n ბიტში. მაგალითისთვის ავიღოთ 3-მეტობის კოდირება. ამ სქემაში ‘0’ ბიტი კოდირდება, როგორც ‘000’, ხოლო ‘1’ ბიტი კი როგორც



‘111’. არხის დეკოდერი ბიტების n ბლოკებიდან ახდენს მონაცემების გენერაციას.

დეკოდერი იღებს n ბიტიან ბლოკებს და ელოდება, რომ ყველა ცალკეულ ბლოკში ყველა ბიტს აქვს ერთნაირი მნიშვნელობა. წინააღმდეგ შემთხვევაში დეკოდერი ადგენს შეცდომას. ამ დროს ზოგიერთო შეცდომა შესაძლოა გასწორდეს. მაგალითად 3-სიჭარბის კოდირებაში გადასაცემია ‘000’.

მიღებული	000	010	100	101	111
დეკოდირებული	0	0	0	1	1

5-სიჭარბის კოდირებაში გასაცემია ‘00000’.

მიღებული	00000	00010	01001	01011	10111
დეკოდირებელი	0	0	0	1	1

როგორც ვხედავთ 3-სიჭარბის კოდირებას შეუძლია 1 ბიტის კორექცია დაშიფრულ სიტყვაში, ხოლო 5-სიჭარბის კოდირებას შეუძლია 2 ბიტის კორექცია.

ძირითადი ფორმულა: n-სიჭარბის კოდირებას შეუძლია (n-1)/2 ბიტამდე კორექცია კოდში.

რაც უფრო დიდია n-ის მნიშვნელობა, მით უფრო მეტია კორექციის შესაძლებლობები. რა გვიშლის ხელს 1000000-სიჭარბის კოდირების გამოყენებაში? მიზეზი ოპერაციის დროულად შესრულებაშია. დავუშვათ საკომუნიკაციო სისტემა მუშაობს 100 კილო ბიტ წამში(kbps) სიჩქარით. თუ გადასაცემი მონაცემია 10 ბიტი, არაკოდირებული სიგნალი გადაეცემა 10/100=0,1 წმ-ში. კოდირებული სიგნალი ბიტების უფრო დიდ რაოდენობას შეიცავს, შესაბამისად ოპერაციის დროც უფრო მეტი იქნება.

არხში გადაცემის სიჩქარის გაზრდითა და ჭარბი კოდირების ერთად, კომბინირებულად გამოყენებით შესაძლებელია მნიშვნელოვნად შევამციროთ შეცდომების ალბათობა. მაგალითად თუ ორიგინალურ(არაკოდირებულ) სისტემაშ დყოვნება 1მწმ-ია, მაშინ ორიგინალური სისტემის შედეგია ბიტების 30%-იანი შეცდომა(BER). შესაძლებელია გამოვიყენოთ დაშიფვრა რათა შევამციროთ BER 5%-მდე დროის დაყოვნების 10მწმ-მდე გაზრდით, ან 0,2%-მდე დაყოვნების 2წმ-მდე გაზრდით. თუ რამხელა უნდა იყოს დაყოვნება ინჟინერ-კონსტრუქტორები წყვეტენ ცალკეული დანართისთვის ინდივიდუალურად. მის ხასიათსა და მგრძნობიარობაზე დაყრდნობით. მაგალითისთვის 30წმ-იანი დაყოვნება მისაღებია მეილის გაგზავნისას ან ფაილი ჩამოტვირთვის დროს, მაგრამ სრულიად მიუღებელია მოლაპარაკეთა მხარეებს შორის. განსაკუთრებით მაშინ, როცა განსაკუთრებული სიტუაცია წარმოიქმნება.

პრაქტიკულად ყველა ბლოკური კოდი წრფივია. ეს დაკავშირებულია იმასთან, რომ არაწრფივი კოდების გამოკვლევა გაცილებით რთულია და მათვის ძნელია უზრუნველვყოთ კოდირების და დეკოდირების მისაღები სიადვილე. წრფივი ბლოკური კოდი - ისეთი კოდია, რომ კოდური სიტვების სიმრავლე ქმნის *k*-განზომილებიან წრფივ ქვესივრცეს ქმნის(დავარქვათ მას C) *n*-განზომილებიან წრფივ სივრცეში, რომელიც *k*-ბიტური ვექტორების სივრცის იზომორფულია.

ეს ნიშანვს, რომ კოდირების ოპერაცია შეესაბამება საწყისი *k*-ბიტური ვექტორის გამრავლებას არაგადაგვარებულ G მატრიცაზე. დავუშვათ $C^{\perp}$ ორთოგონალური ქვესივრცეა C-თან მიმართებაში, ხოლო H- ამ ქვესივრცის ბაზისური მატრიცა. მაშინ ნებისმიერი ვექტორისთვის $\overrightarrow{v} \in C$ სამართლიანია:

$$\overrightarrow{v} H^T = \overrightarrow{0}.$$

წრფივი კოდების დეკოდირების საერთო მეთოდი

ნებისმისერი კოდის(მათ შორის არაწრფივიც) დეკოდირება შესაძლებელია ჩვეულებრივი ცხრილით, სადაც ყოველ მიღებულ $\overrightarrow{r}_i$ სიტყვას შეესაბამება ყველაზე მეტად სავარაუდო სიტყვა $\overrightarrow{u}_i$. თუმცა ეს მეთოდი დიდი ცხრილების გამოყენებას მოითხოვს შედარებით მცირე სიგრძის კოდირების სიტვებისთვის.

წრფივი კოდებისთვის შესაძლებელია მისი გამარტივება. ამ დროს ნებისმიერი მიღებული $\vec{r}_i$ ვექტორისთვის გამოითვლება $\vec{s}_i = \vec{r}_i H^T$. რახან $\vec{r}_i = \vec{v}_i + \vec{e}_i$ სადაც $\vec{v}_i$ - კოდურის სიტყვაა, ხოლო $\vec{e}_i$-შეცდომების ვექტორი, გამოდის: $\vec{s}_i = \vec{e}_i H^T$. შემდეგ მის მიხედვით ცხრილში განისაზღვრება შეცდომის ვექტორი, რომლის მიხედვითაც განისაზღვრება გადაცემული კოდური სიტყვა. ამ დროს ცხრილი გაცილებით პატარა გამოდის წინა მეთოდისგნა განსხვავებით.

ციკლური წრფივი კოდები

ციკლური კოდი - წრფივი კოდი, რომელსაც ციკლური თვისება გააჩნია. ანუ კოდური სიტყვის ტოველი ციკლური გადალაგება ასევე წარმოადგენს კოდურ სიტყვას. დავუშვათ, რომ $\overline{y} = (y_0, y_1, \ldots, y_{n-1}) \in \mathbb{GF}(q^n)$ n სიგძის სიტყვაა ანბანიდან, რომლსი ელემენტების სასრული სასრული არიდან $\mathbb{GF}(q)$ და $y(x) = y_0 + y_1 x + y_2 x^2 + \ldots + y_{n-1} x^{n-1}$ პოლინომი, რომელიც ამ სიტყვას შეესაბამება. როგორც ვხედავთ შესაბამისობა არამარტო ურთიერთ ცალსახაა, არამედ იზომორფულიც. რახან „სიტყვები“ შედგებიან სასრული არის ასებისგან, მაშინ შესაძლებელია მათი შეკრება და გამრავლება, ამასთან შედეგი იმავე არეში იქნება. პოლინომი, რომელიც შეუსაბამებს წრფივ კომბინაციას $\overline{y} = m_1\overline{y_1} + m_2\overline{y_2}$ სიტყვათა წყვილი $\overline{y_1} = (y_{1,0}, \ldots, y_{1,n-1})$ და $\overline{y_2} = (y_{2,0}, \ldots, y_{2,n-1})$, ტოლია ამ სიტყვების პოლინომების წრფივი კომბინაციის:

$$\overline{y}(x) = \sum_{k=0}^{n-1}(m_1 y_{1k} + m_2 y_{2k})x^k = m_1\overline{y_1}(x) + m_2\overline{y_2}(x)$$

ეს საშუალებას გვაძლევს n სიგრძის სიტყვათა სიმრავლე განვიხილოთ, როგორც პოლინომების წრფივი სივრცე არაუმეტეს n-1 ხარისხით.

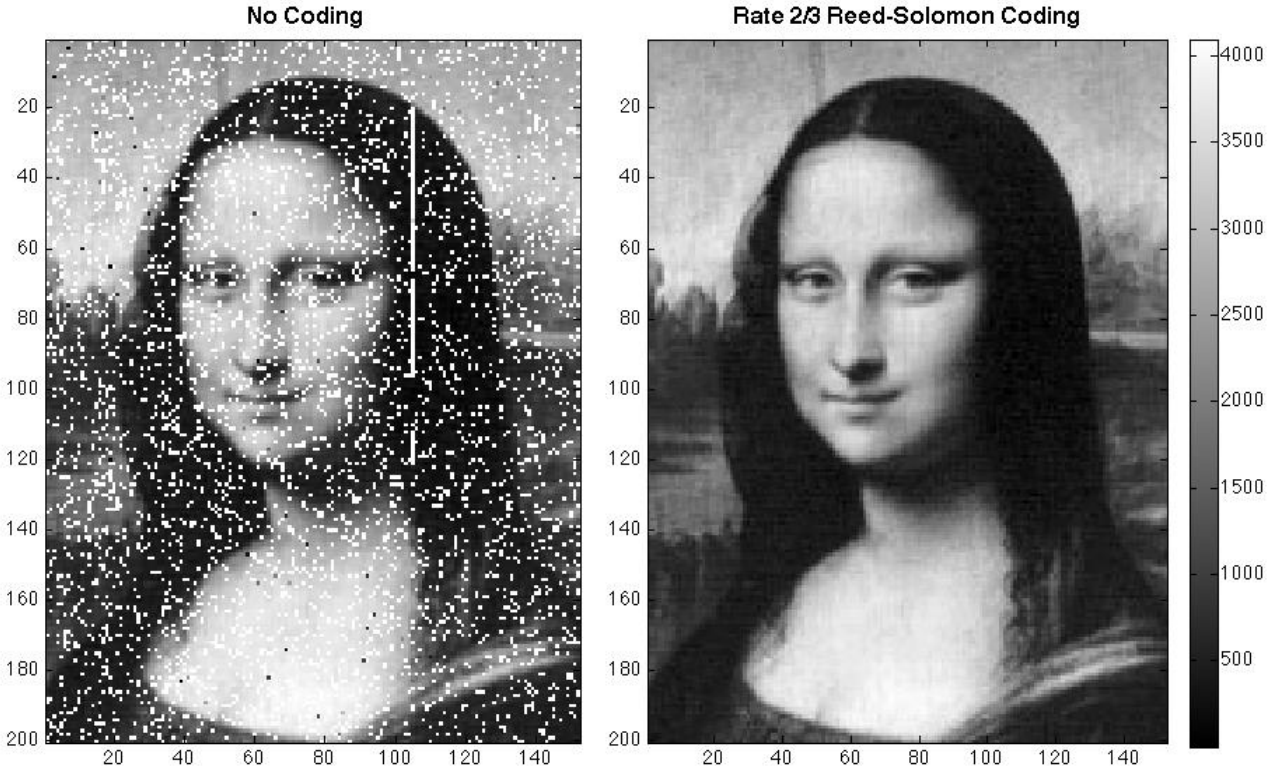
მიუხედავად იმისა, რომ წრფივი კოდების დეკოდირება უფრო ადვილია ვიდრე უმეტესი არაწრფივი კოდისს, ეს ამოცანა მაინც საკმარისად რთული რჩება. წრფივი ციკლური კოდები ამას მნიშვნელოვნად ამარტივებენ.

მიუხედავათ იმისა, რომ ბლოკური კოდები კარგად უმკლავდებიან იშვიათ, მაგრამ დიდ დასტა შეცდომებს, მათი ეფექტურობა ხშირ, მაგრამ მცირე შეცდომების დროს შედარებით მცირეა.

ციკლური კოდების მაკალითებია:

- BCH კოდი. მისი განმასხვავებელი თვისებაა ისაა, რომ შესაძლებელია ის ავაწყოთ ისე, რომ მინიმალური მანძილი იყოს მოცემულზე არანაკლები.

- *Reed–Solomon* კოდი. კოდური ვექტორის ელემენტებს წარმოადგენს არა ბიტები, არამედ ბიტების ჯგუფები(ბლოკები). ძალიან გავრცელებულია რიდი-სოლომონის კოდები, რომლებიც ბაიტებთან მუშაობს(ოქტეტებთან). მათემატიკურად რიდი-სოლომონის კოდები წარმოადგენს BCH კოდს.



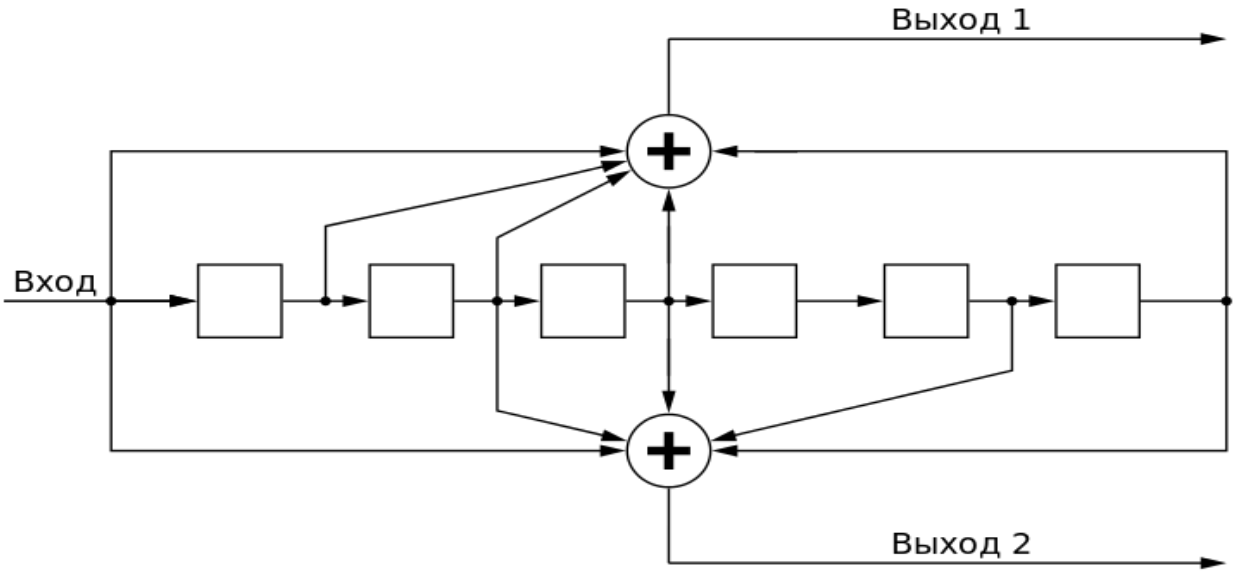
კონვოლუციური კოდირება

კონვოლუციური კოდი, ბლოკურისგან განსხვავებით არ ყოფს ინფორმაციას ფრაგმენტებად, არამედ მუშაობს მასთან როგორც უწყვეტ ნაკადთან. მისი კოდირება ძალიან მარტივი სამუშაოა, რასაც ვერ ვიტყვით მის დეკოდირებაზე. კოდირება კონვოლუციური კოდით ხდება რეგისტრული წნაცვლების საშუალებით. დეკოდირება კი ძირითადად ვიტერბის ალგორითმით ხდება, რომელიც ცდილობს აღადგინოს გადაცემული მიმდევრობა მაქსიმალური დამაჯერებლობის კრიტერიუმის მიხედვით.

კონვლოუციური კოდი ეფექტურად მუშაობს არხში თეთრი ხმაურით, მაგრამ ძნელად უმკლავდება შეცდომათა პაკეტებს. უფრო მეტიც, თუ დეკოდერი შეცდა, გამომავალი შედეგიც შეიცავს შეცდომათა პაკეტს.

არხის კოდირების კიდევ ერთი მაგალითია კონვოლუციური კოდირება. ის საშუალებას გვაძლევს უფროხარისხიანი კორექცია ვაწარმოოთ, ვიდრე n-ჭარბობის

კოდირებაში. m/n კონვოლუციური დამშიფრავი თანმიმდევრული სისტემაა, რომელიც აგენერირებს n-ბიტიან სიტყვას m ბიტიდან. სიგნალი გენერირდება მრავალწევრების გამოყენებით და ორობითი შეკრებით.



კონვოლუციური კოდერი ($k = 7,\ R = 1/2$)

კოდერის ყოველ სამუშაო ტაქტზე შემავალი k სიმბოლო ნახევრად უსასრულო მიმდევრობიდან გარდაიქმნება $n > k$ გამომავალ სიმბოლოში. გარდაქმნაში ასევე მონაწილეობს წინა m სიმბოლოც. სრულდება წრფივობის თვისება. ამგვარი კოდირება აღმოაჩინა 1959 წელს ჰეგელბერგმა(*Hegelbeger*).

განვიხილოთ მაგალითი. 1/3 დამშიფრავი შემდეგი მრავაწევრებით:

$$G_1 = (1,0,1),\ G_2 = (0,1,1),\ G_3 = (1,1,0).$$

$$y_1(i) = 1 \times x(i) \oplus 0 \times x(i-1) \oplus 1 \times x(i-2) \qquad G_1$$
$$y_2(i) = 0 \times x(i) \oplus 1 \times x(i-1) \oplus 1 \times x(i-2) \qquad G_2$$
$$y_3(i) = 1 \times x(i) \oplus 1 \times x(i-1) \oplus 0 \times x(i-2) \qquad G_3$$

ყოველი x(i) ბიტი შეესაბამება 3-ბიტიან კოდს $y_1(i)y_2(i)y_3(i)$, სადაც:

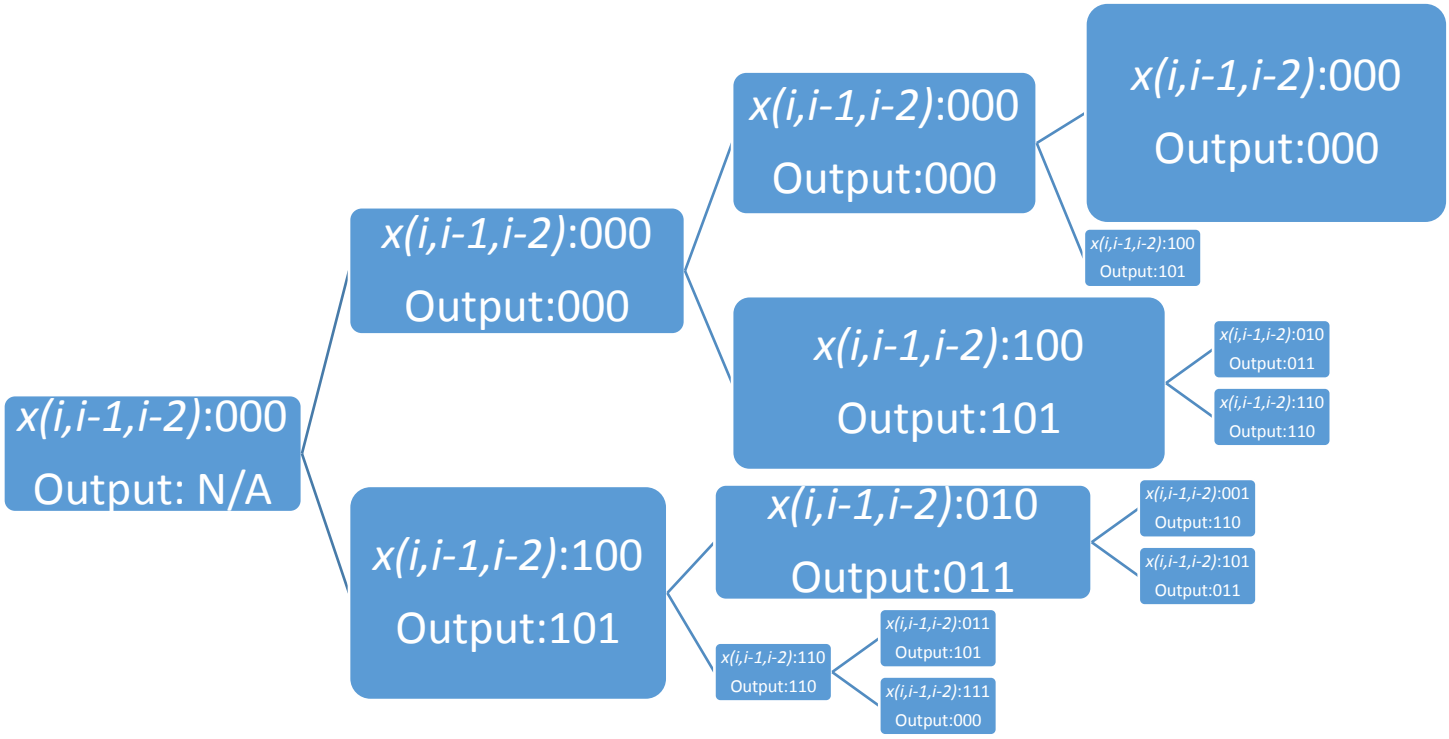
x(*i*), *x*(*i-1*), და *x*(*i-2*) წარმოადგენენ ახლანდელს, წინანდელს და დაყოვნების შემომავალ პარამეტრებს. დავუშვათ სისტემა თავდაპირველად დარესეტებულია ‘0’-ზე. x(i-1), x(i-2) თავდაპირველი მნიშვნელობებია 0,0. თუ ბიტი *x*(*i*)=0, გამომავალია y=000, თუ ბიტი *x*(*i*)=1, გამომავალია y=101.

მეორე ბიტისთვის:

- თუ პირველი ბიტი იყო 1, x(i-1), x(i-2) იქნებიან 1,0.
  - თუ მეორე ბიტი x(i)=0, გამომავალი იქნება y=011
  - თუ მეორე ბიტი x(i)=1, გამომავალი იქნება y=110

- თუ პირველი ბიტი იყო 0, x(i-1), x(i-2) არის 0,0
  - თუ მეორე ბიტი x(i)=0, გამომავალი იქნება y=000
  - თუ მეორე ბიტი x(i)=1, გამომავალი იქნება y=101

ეს მიდგომა შეიძლება დიაგრამის ასაგებად გამოვიყენოთ(იხ. დიაგ. 1). რახან ყოველი კოდირებული სიტყვაშექმნილის სამი ბიტისგან x(i,i-1,i-2), დეკოდერი კითხულობს



დიაგ. 1

კოდირებულ სიტყვას რომელიც წარმოდგენილია 3 ბიტიანი ბლოკებისგან(იხ. ცხ. 1).

კონვოლუციური კოდერის სტრუქტურა: ყოველ დამშიფრავს შეესაბამება სხავადასვა

ხის დიაგრამა. კოდირების სხვადახვა კოეფიციენტები და გენერატორის სხავადსხვა პოლინომები შედეგად გვაძლევს სხავადსხვა მაკორექტირებელ ეფექტს. კონვოლუციური კოდერი 2 ან მეტი ვარგისი კოდის მიმდევრობით, რომელიც გაყოფილია 1 ბიტით არის წმინდა კოდერი. უკეთესია კონვოლუციური კოდერი სადაც ვარგისი კოდის მიმდევრობა მნიშვნელოვნად დაყოფილია/განსხვავებულია ერთმანეთისგან.

კოდის სწორი მიმდევიბა	შეუსაბამობა
000 000 000	5
000 000 101	7
000 101 011	5
000 101 110	5
101 011 110	5
101 011 011	5
101 110 101	5
101 110 000	3

მინიმალური შეუსაბამობა

მიღებული კოდი:
110 110 010

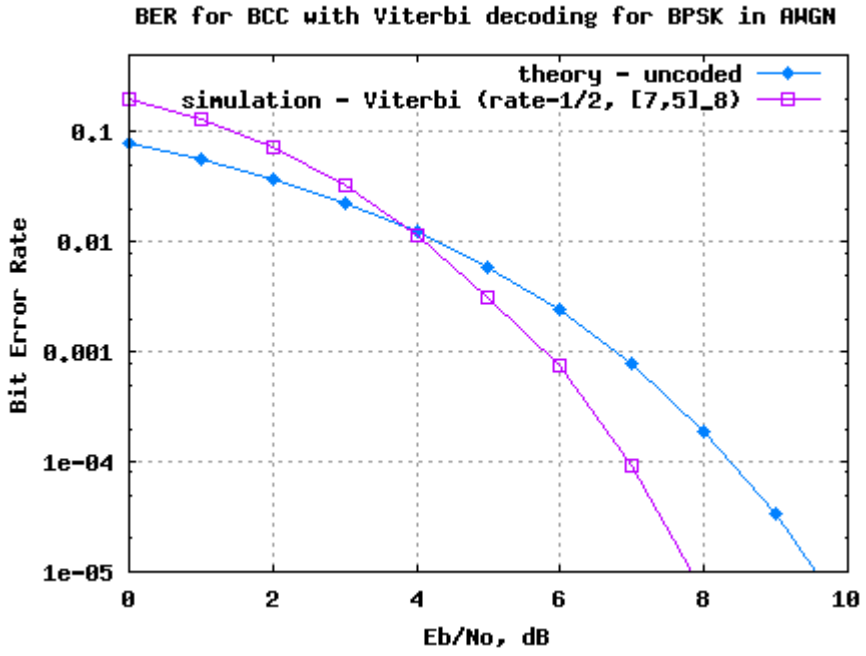
ცხ. 1

ზემოთ აღწერილი არხების კოდირების სიმულაციას ვახდენთ Matlab-ის დახმარებით. ვახდენთ BER მწამოებლობის და *n* კოფიციენტის დამოკიდებულების გრაფიკის შექმნას. ასევე დამუშავების დროის და n კოეფიციენტის დამოკიდებულების გრაფიკის შექმნას.

ვიტერბის ალგორითმი

ვიტერბის ალგორითმი გამოიყენება კონვოლუციური კოდის დეკოდირებისთვის. ალგორითმი შეთავაზებული იქნა ანდერი ვიტერბის მიერ 1967 წელს და გარდა კონვოლუციურ კოდირებაში ის ფართოდ გამოიყენება ხმის გამოცნობაში, ხმის სინთეზში, კომპიუტერულ ლინგვისტიკაში და ბიოინფორმატიკაში. ალგორითმი ახდენს რამოდენიმე ვარაუდის შეტავაზებას:

- დაკვირვებადი და ფარული მოვლენები მიმდევრობები უნდა იყოს. მიმდევრობა ხშირად მოწესრიგებულია დროის მიხედვით.
- ორი მიმდევრობა გასწორებული უნდა იყოს: ყოველი დაკვირვებადი მოვლენა შეესაბამება ზუსტად ერთ ფარულ მოვლენას.
- ყველაზე სავარაუდო ფარული მიმდევრობის გამოთვლა t მომენტამდე დამოკიდებული უნდა იყოს მხოლოდ დაკვირვებად მოვლენაზე t დროის მომენტში და ყველაზე სავარაუდო მიმდევრობაზე t−1 მომენტამდე.



დავუშვათ ჩვენ გვაქვს ფარული მარკოვის მოდელი S მდგომარეობების სივრცით, საწყისი ალბათობებით π_i, რომლებიც i მდგომარეობაშია, და ალბატობებით $a_{i,j}$, i-ური მდგომარეობიდან გადასვლით j-ურში. დავუშვათ გამომავალზე ვაკვირდებით $y_1, \ldots, y_T$. მაშინ მდგომარობათა ყველაზე სავარაუდო მიმდევრობა $x_1, \ldots, x_T$ მოიცემა რეკურენტული მიმართებით:

$$\begin{aligned} V_{1,k} &= \mathrm{P}\left(y_1 \mid k\right) \cdot \pi_k \\ V_{t,k} &= \mathrm{P}\left(y_t \mid k\right) \cdot \max_{x \in S} \left(a_{x,k} \cdot V_{t-1,x}\right) \end{aligned}$$

აქ $V_{t,k}$ არის მდგომარეობათა ყველაზე სავარაუდო მიმდევრობა, რომელიც პასუხისმგებელია პირველი t დაკვირვებადი სიმბოლოს გამოჩენაზე, რომლებიც მთავრდება k მდგომარეობაში. ვიტერბის გზა შეიძლება ნაპოვნი იყოს მიმტითებლების მეშვეობით, რომლებიც იმახსოვრებენ რა x მდგომარეობა გამოჩნდა მეორე ტოლობაში. დავუშვათ $\mathrm{Ptr}(k,t)$ არის ფუნქცია, რომელიც აბრუნებს x-ს, $V_{t,k}$ -ს დასათვლელად გამოყენებულს, თუ $t>1$ და k-ს თუ $t=1$. მაშინ:

$$\begin{aligned} x_T &= \arg\max_{x\in S}(V_{T,x}) \\ x_{t-1} &= \mathrm{Ptr}(x_t,t) \end{aligned}$$

ამ ალგორითმის სირთულეა $O(T\times|S|^2)$.

ხელახალი გადაცემის ავტომატური მოთხოვნა

ხელახალი გადაცემის ავტომატური მოთხვნის სისტემები(ARQ — Automatic Repeat reQuest) დაფუძნებულია შეცდომების აღმომჩენ კოდებზე. გავრცელებულია ავტომატური მოთხოვნის შემდეგი სახეები:

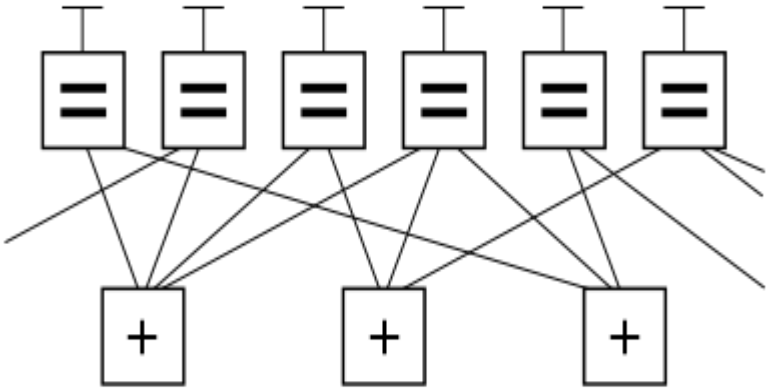
- ARQ მოთხოვნა დაყოვნებით(stop-and-wait ARQ). ამ მეთოდის იდეა მდგომარეობს იმაში, რომ გადამცემი ელოდება მიმღებისგან დასტურს ინფორმაციის წარმატებით მიღების შესახებ ახალი მონაცემების გადაგზავნამდე. თუ მონაცემთა ბლოკი შეცდომით გადაიცა, მიმღები აგზავნის უარყოფით დასტურს(negative acknowledgment, NAK) და გადამცემი იმეორებს ბლოკის გაგზავნას. ეს მეთოდი მისაღებია ნახევრადუპლექსური საკომუნიკაციო არხისთვის. მისი ნაკლია დაბალი სიჩქარე.
- შეუწყვეტელი ARQ მოთხოვნა დაბრუნებით(continuous ARQ with pullback). მისთვის საჭიროა ნახევრადდუპლექსური საკომუნიკაციო არხი. მონაცემთა გაგზავნა გადამცემიდან მიმღებზე ერთდოულად ხდება. შეცდომის შემთხვევაში გაგზავნა განახლდება შეცდომიანი ბლოკიდან.
- შეუწყვეტელი ARQ მოთხოვნა ამორჩევითი განმეორებით(continuous ARQ with selective repeate). ამ დროს ხდება მხოლოდ დაზიანებული ბლოკების გადაცემა

კასკადური კოდირება, იტერაციული დეკოდირება

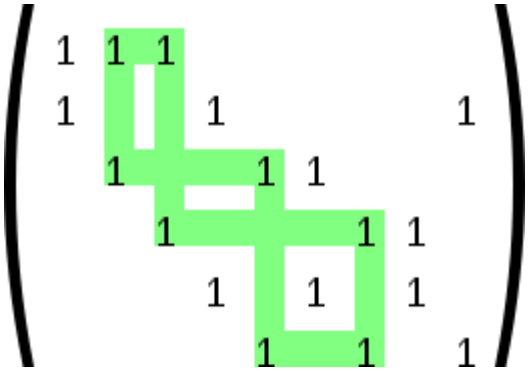
კოდირების სხვადახსვა საშუალებების უპირატესობები შეიძლება გავაერთიანოთ კასკადური კოდირების გამოყენებით. ამ დრო ინფორმაცია ჯერ კოდირდება ერთი კოდით, შემდეგ კი მეორეთი. მიიღება კოდი-პროდუქტი.

მაგალითად პოპულარულია შემდეგი კონსტრუქცია: მონაცემები კოდირდება რიდი-სოლომონის კოდით, შემდეგ გაიფანტება(სიმბოლეოები რომლებიც ერთმანეთთან ახლოს იყო განლაგდება ერთმანეთის მოშორიებით) და კოდირდება კონვოლუციური კოდით. მიმღებზე ჯერ დეკოდირდება კონვოლუციური კოდი შემდეგ ხდება გაფანტვის საწინააღმდეგო ქმედება და შემდეგ ხდება რიდი-სოლომონის დეკოდირება.

ზოგიერთი კოდი-პროდუქტი სპეციალურადაა კონსრუირებული იტერაციული დეკოდირებისთვის, რომლის დროსაც დეკოდირება ხდება რამოდენიმე მისვლით. ყოველი მათგანი იყენებს ინფორმაციას წინაზე. ეს ზრდის ეფექტურობას, თუმცა დეკოდირება უფრო მეტ რესურს მოითხოვს. ასეთ კოდებს მიეკუთვნება ტურბო კოდები და LDPC კოდები(გალაგერის კოდები).



LDPC კოდის წარმოდგენა გრაფით



LDPC კოდის შემმოწმებელი მატრიცა.

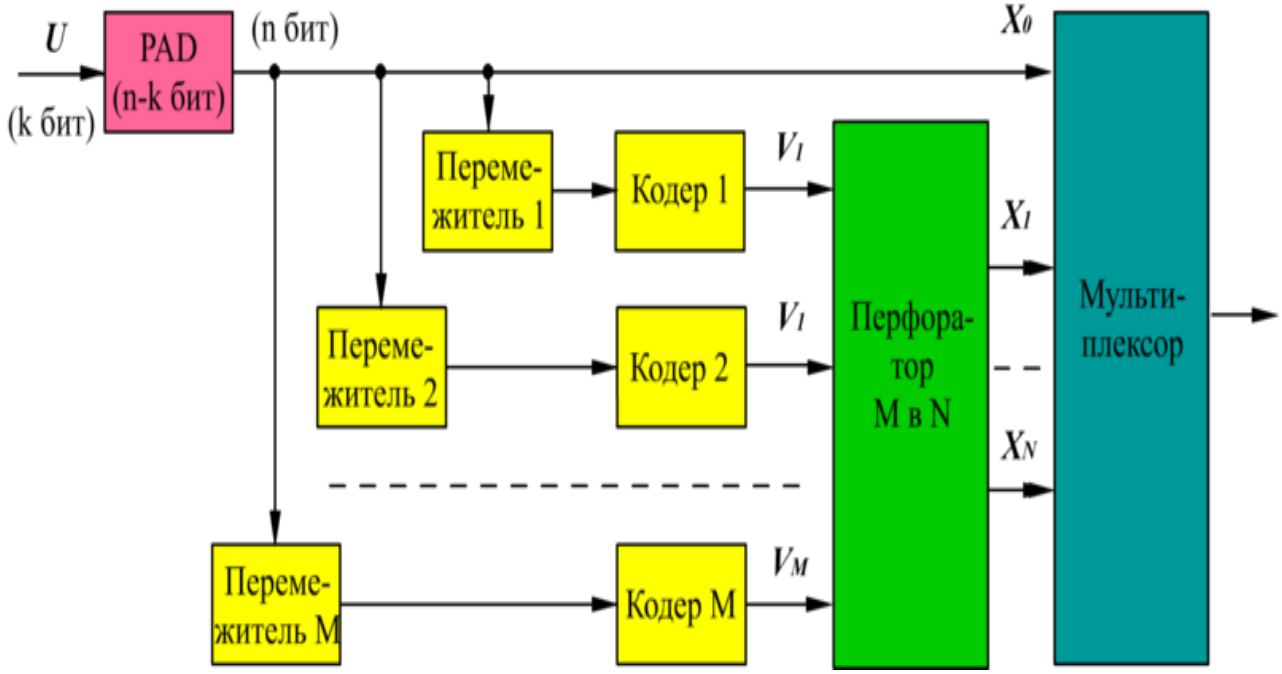
კოდის ეფექტურობა განისაზღვრება შეცდომების რაოდენობით, რომლების გასწორებაც მას შეუძლია, ზედმეტი ინფორმაციის რაოდენობით, რომლის დამატებაც

საჭიროა, და ასევე კოდირების და დეკოდირების რეალიზაციის სირთულით(როგორც აპარატული ასევე როგორც პროგრამა). ვთქვათ გვაქვს ორობითი ბლოკური კოდი *(n, k) t* მაკორექტირებელი თვისებით. მაშინ სამართლიანია ტოლობა:

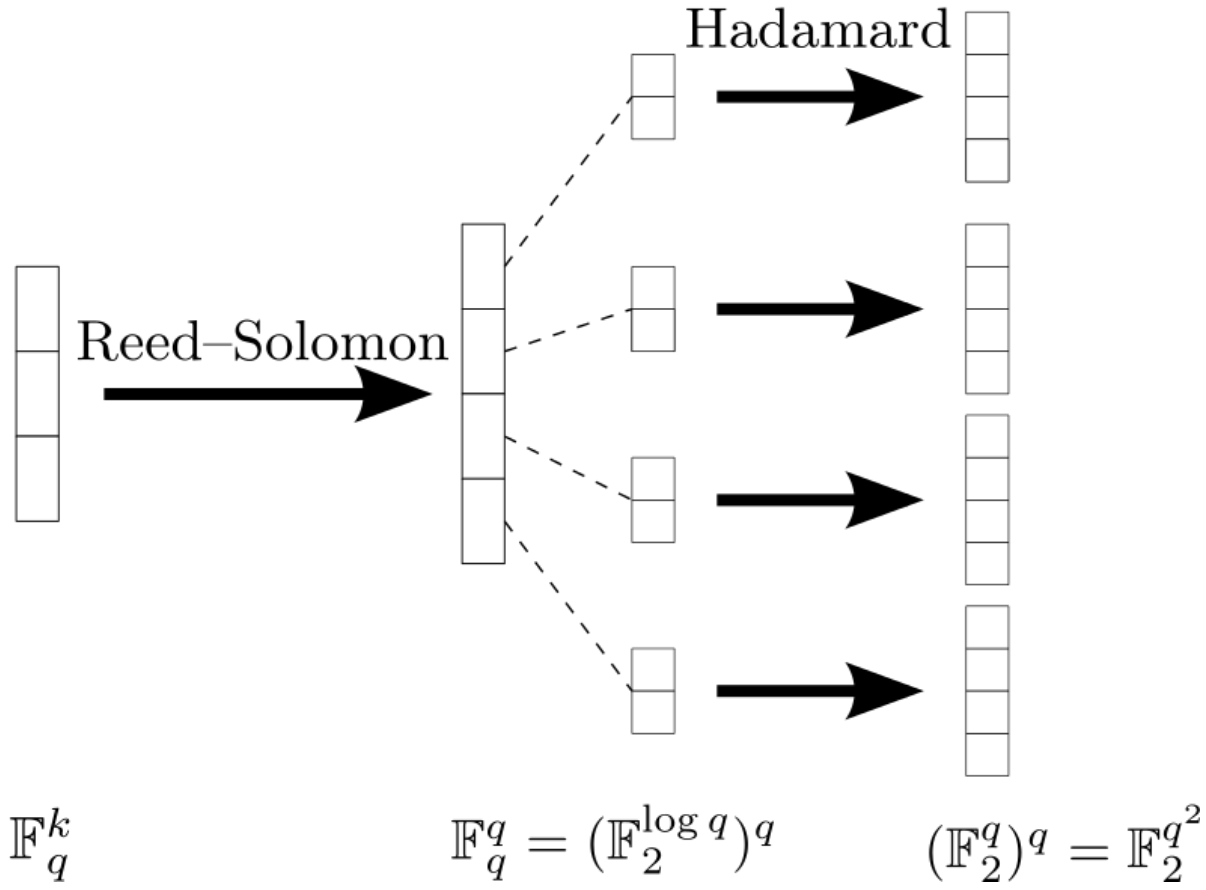
$$\sum_{i=0}^{t} \binom{n}{i} \leqslant 2^{n-k}.$$

კოდები რომლებიც აკმაყოფილებს ამ ტოლობას სრულყოფილი ეწოდება. სრულყოფილ კოდებს მაგალითისთვის მიეკუთვნება ჰემინგის კოდები. პრაქტიკაში ხშირად გამოყენებული დიდი მაკორეტირებელი თვისების მქონე კოდები(მაგ. რიდი-სოლომონის კოდი) სრულყოფილები არ არიან.

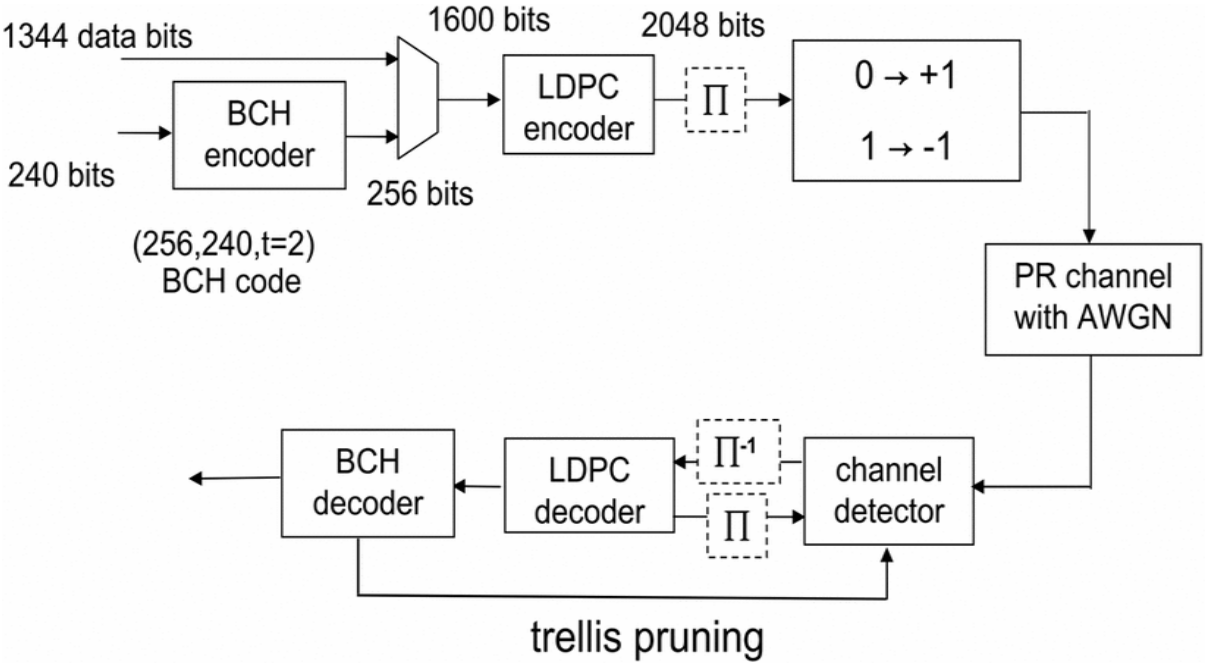
ინფორმაციის გადაცემისას საკომუნიკაციო არხში შეცდომის ალბატობა დამოკიდებულია *SNR*-ზე დემოდულატორის შესასვლელზე. ამგვარად ხმაურის მუდმივი დონის დროს გადამწყვეტია გადამცემის სიმძლავრე. მობილურ და თანამგზავრულ სისტემებში მწვავედ დგას ენერგიის ეკონომიის საკითხი. არხის კოდირება საშუალებას გვაძლევს შეცდომები გავასწოროთ ისე რომ, გადამცემის სიმძლავრე შევამციროთ, ხოლო ინფორმაციის გადაცემის სიჩქარე უცვლელი დავტოვოთ.



ტურბო კოდის საერთო სტრუქტურული სქემა



რიდი-სოლომონის კოდის და ჰადამარდის კოდის კონკეტენაცია



დასკვნა

BER წარმოადგენს პარამეტრს, რომელიც გადამცემი არხის მწარმოებლობის მშვენიერ ინდიკაციის საშუალებას გვაძლევს. ნებისმიერი არხის ყველაზე მთავარ პარამეტრს წამოადგენს შეცდომების რაოდენობა, რომელიც წარმოიქმნება გადაცემის დროს. მისი ცოდნა ასევე გვიაქტიურებს სხვა საშუალებებს როგორებიცაა სიმძლავრე და გამტარუნარიანობა და ა.შ ადაპტირებული იყოს საჭირო წარმადობაზე. BER ტესტინგი მძლავრი საშუალებაა გადამცემი სისტემების საბოლოოდ შემოწმებისთვის. თუ BER ძალიან მაღალია მაშინ სისტემის წარმადობა უარესდება. იმ შემთხვევაში, როცა ის გარკვეულ ფარგლებშია მოქცეული, სისტემა დამაკმაყოფილებლად მუშაობს. ამიტომ მნიშვნელოვანი იყო არა მარტო გამოგვეთვალა სისტემის BER-ის მნიშვნელობები და ისინი გრაფიკზე აგვესახა. არამედ გვეპოვა გზები ამ მონაცემის შესამცირებლად არხის კოდირების სხვადასვა საშუალებების გამოყენებით.

გამოყენებული ლიტერატურა

JAMES E. GILLEY: “BIT-ERROR-RATE SIMULATION USING MAT LAB”, TRANSCRYPT INTERNATIONAL, INC., AUGUST 19, 2003.

WIKIPEDIA, FREE ENCYCLOPAEDIA, ARTICLE ON BIT ERROR RATE HTTP://EN.WIKIPEDIA.ORG/WIKI/BIT_ERROR_RATE.

WIKIPEDIA, FREE ENCYCLOPAEDIA, ARTICLE ON SIGNAL TO NOISE RATIO HTTP://EN.WIKIPEDIA.ORG/WIKI/S/N_RATIO

JOHN. G. PROAKIS, “DIGITAL COMMUNICATIONS”, MCGRAW-HILL SERIES IN ELECTRICAL AND COMPUTER ENGINEERING, THIRD ED.

THE MATH WORKS, INC., THE STUDENT EDITION OF MATLAB VERSION 7 USER'S GUIDE, PRENTICE HALL, ISBN 0-13-184979-4, 1995.

D. HANSEL MAN AND B. LITTLEFIELD, MASTERING MATLAB 7. A COMPREHENSIVE TUTORIAL AND REFERENCE, PRENTICE HALL, UPPER SADDLE RIVER, NJ, 1998

B. SKLAR, DIGITAL COMMUNICATIONS: FUNDAMENTALS AND APPLICATIONS, CH. 4, ENGLEWOOD CLIFFS, NJ: PRENTICE HALL, 1988.

M. JERUCHIM, “TECHNIQUES FOR ESTIMATING THE BIT ERROR RATE IN THE SIMULATION OF DIGITAL COMMUNICATION SYSTEMS," IEEE J. SELECT. AREAS COMMUNICATION., VOL. SAC-2, PP. 153-

WIKIPEDIA , Error detection and correction

https://en.wikipedia.org/wiki/Error_detection_and_correction