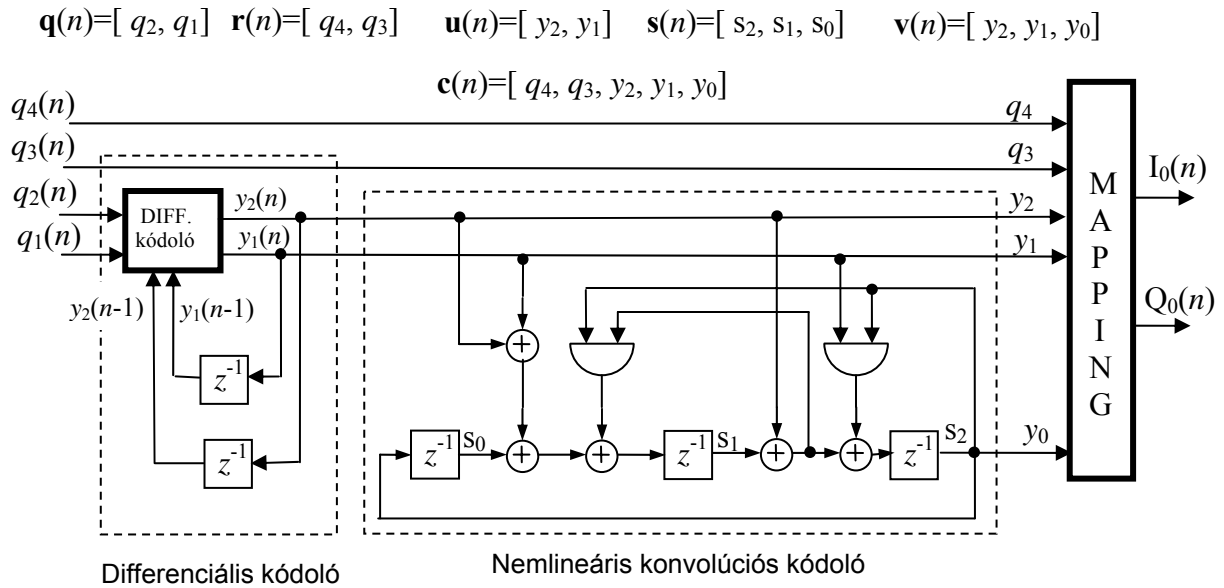


### 15.4. A trellis-kód moduláció (TCM)

A trellis-kód moduláció olyan digitális moduláció, melynél a **konvolúciós kódolással** előállított kódokat ( $\mathbf{c}$ ) úgy **képezzük le a kvadratúra jeltérbe** ( $I_0, Q_0$ ), hogy a dekóderben az additív zaj következtében fellépő döntési hiba valószínűségét minimalizáljuk.

A téma tárgyalását egy példa kapcsán fogjuk végezni. Az ITU (*International Telecommunication Union*) V.17.-es ajánlása remek lehetőséget kínál arra, hogy a trellis-kód modulációnál felmerülő kérdésekkel megismerkedjünk.

A 15.8.ábrán az ajánlásban szereplő kódoló tömbvázlatát látjuk:



15.8. ábra Az ITU V.17.ajánlás konvolúciós kódolója

A kódolóban láthatóan három fő egységet különböztethetünk meg:

- a differenciális kódolót,
- a konvolúciós kódolót és a
- *mapper*-t, ami a kódnak a jeltérbe történő leképezését végzi.

A differenciális kódoló szigorúan véve nem tartozik a trellis-kód moduláció kérdésköréhez, erre az egységre a QAM demodulátor vivő-visszaállításának sajátossága miatt van szükség. Mégis azért tárgyaljuk itt ezt a kérdést is, mert érinti az egyes kódok leképzési szabályát.

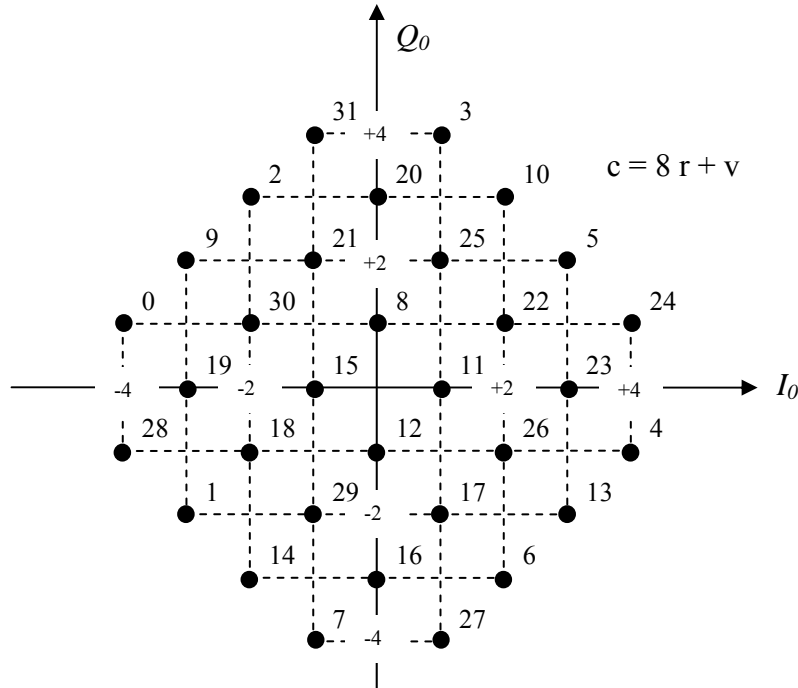
Az ábrában feltüntettük azokat a jelöléseket, melyek segítségével bizonyos bit-csoportokra hivatkozni fogunk. A biteket egy (helyértékkel rendelkező) bináris vektorba rendeztük. Az így kapott bináris számot ezután egy decimális számként kezeljük

Az  $n$ -ik időrésben a kódoló bemenetére 4 bit érkezik:  $q_4(n)$ ,  $q_3(n)$ ,  $q_2(n)$ ,  $q_1(n)$ . Ezeket két csoportra osztottuk, a differenciálisan kódolt  $\mathbf{q}(n)=[q_2, q_1]$  és a differenciális kódolásban részt nem vevő  $\mathbf{r}(n)=[q_4, q_3]$  bitekre. A differenciális kódoló kimenetei az  $\mathbf{u}(n)=[y_2, y_1]$  bitek. A konvolúciós kódoló állapotát (state) az  $\mathbf{s}(n)=[s_2, s_1, s_0]$  változóval írjuk le. A konvolúciós kódoló  $\mathbf{v}(n)=[y_2, y_1, y_0]$  kimenetéből és az  $\mathbf{r}(n)$  bitekből szerkesztjük össze a *mapper* bemenetére érkező ötbités  $\mathbf{c}(n)=[q_4, q_3, y_2, y_1, y_0]$  kódszót.

A kód szisztematikus, mert az üzenet  $[q_4, q_3, y_2, y_1]$  bitjei benne vannak a kódszóban, (csak az  $y_0$  az egyetlen redundáns bit).

### 15.5. A kód leképezési szabályáról

Az ajánlás megadja a  $c$  kódnek a kvadratúra jelekbe  $(I_0, Q_0)$  való leképezésének szabályát. Ezt látjuk az alábbi konstellációs diagramon:



15.9. ábra A  $c$  kód leképezése 32 QAM kvadratúra jelekbe

A tengelyeken relatív amplitúdók, a pontok mellett a  $c$  kód értékei vannak feltüntetve. A  $c = 8r + v$  összefüggés a decimális számbázisból következik, ami lehetővé teszi a tömörebb leírást. (A decimális számot binárisba visszaírva a bitek is rendelkezésre állnak.)

A kódolóban a *leképezést* táblázatok használatával végezzük. A  $c$  kóddal címzett táblázatból egyszerűen kiolvassuk a jelek értékeit.

$c$	$I_0$	$Q_0$
0	-4	+1
1	-3	-2
2	-2	+3
3	+1	+4
4	+4	-1
5	+3	+2
6	+2	-3
7	-1	-4

$c$	$I_0$	$Q_0$
8	0	+1
9	-3	+2
10	+2	+3
11	+1	0
12	0	-1
13	+3	-2
14	-2	-3
15	-1	0

$c$	$I_0$	$Q_0$
16	0	-3
17	+1	-2
18	-2	-1
19	-3	0
20	0	+3
21	-1	+2
22	+2	+1
23	+3	0

$c$	$I_0$	$Q_0$
24	+4	+1
25	+1	+2
26	+2	-1
27	+1	-4
28	-4	-1
29	-1	-2
30	-2	+1
31	-1	+4

15.20. Táblázat A  $MAPI(c)$  és a  $MAPQ(c)$  függvények

Felmerülhet bennünk az a kérdés, hogy milyen megfontolással történt a jelek hozzárendelése a kódokhoz? A leképzésnél két szempontot kellett figyelembe venni:

1. a differenciális kódolást, illetve
2. a trellis-kód moduláció szempontjait.

### 1. A differenciális kódolás

Ismeretes, hogy a QPSK és a QAM rendszerekben a vivőhullám fázisát csak  $k \cdot 90^\circ$  bizonytalansággal tudjuk csak helyreállítani a dekóderek vivő-visszaállító fokozataiban (ahol  $k = 0,1,2,3$ ) (lásd a PLL-ről szóló fejezetet). Ez azt jelenti, hogy a vevőben rekonstruált konstelláció még ideális esetben is  $k \cdot 90^\circ$  elfordulással különbözhet az adóban lévőttől. Nyilvánvalóan ez dekódolási hibát okozna, ha nem alkalmaznánk differenciális kódolást.

A V.17.-es ajánlásban a differenciális kódolást egy kétbites összeadó végzi. (Az összeadóban a második helyértékről a harmadik helyértékre történő átvitel egyszerűen lecsordul). A kódoló  $u(n)$  kimenete a korábbi kimenet  $u(n-1)$  és a  $q(n)$  bemenet összege:

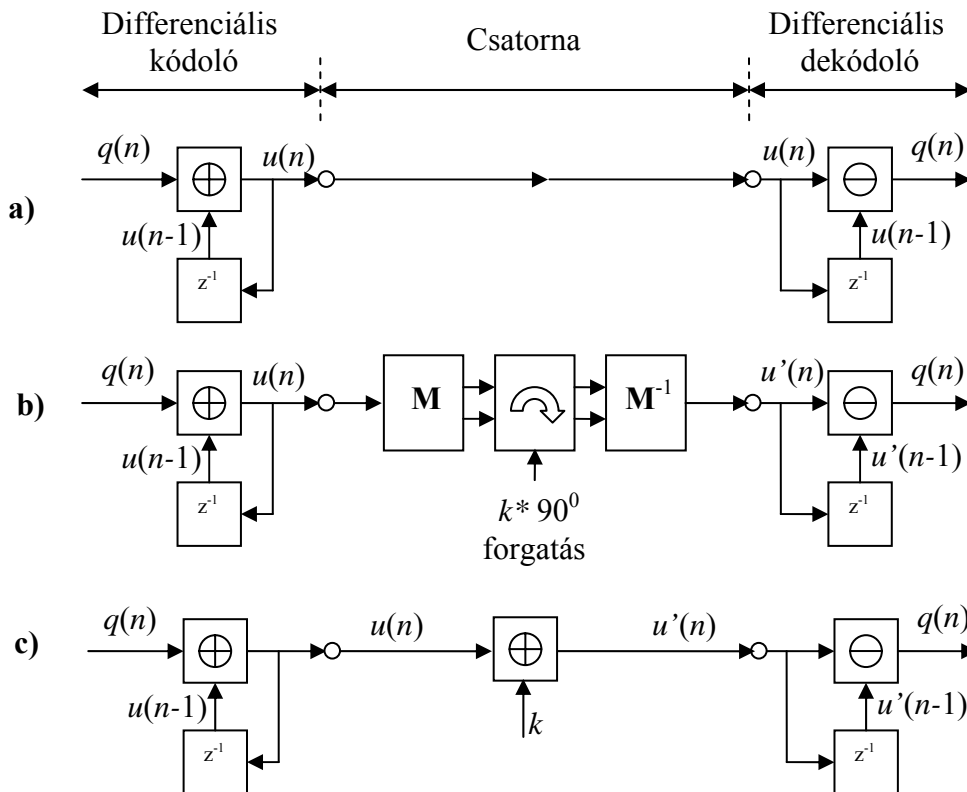
$$u(n) = \{ q(n) + u(n-1) \} \bmod 4 \quad (15.10.)$$

A  $\{ \} \bmod 4$  művelettel az összeadóban a második átvitel elmaradását írjuk le.

A dekódolás a:

$$q(n) = \{ u(n) - u(n-1) \} \bmod 4 \quad (15.11.)$$

művelet elvégzése után hibátlan, amit a 15.10. a.) ábrán is lerajzoltunk.



15.10. ábra A differenciális kódolás és dekódolás folyamata

Tételezzük most fel, hogy elfordulási hiba történt! (Amit a 15.10.b). ábrán próbálunk szemléltetni.) Az  $u(n)$  kódot először az  $M$  eljárással leképeztük kvadratúra jelbe, majd ezt a komplex jelet az óramutató járásával egyezően elforgattuk  $k \cdot 90^\circ$ -al. A leképezés ( $M^{-1}$ ) inverzét (a döntést) végrehajtva  $u'(n)$  kódot kapunk.

Az előzőekben használt differenciális dekóder (kétbites kivonó) akkor ad helyes eredményt, ha teljesül az alábbi összefüggés:

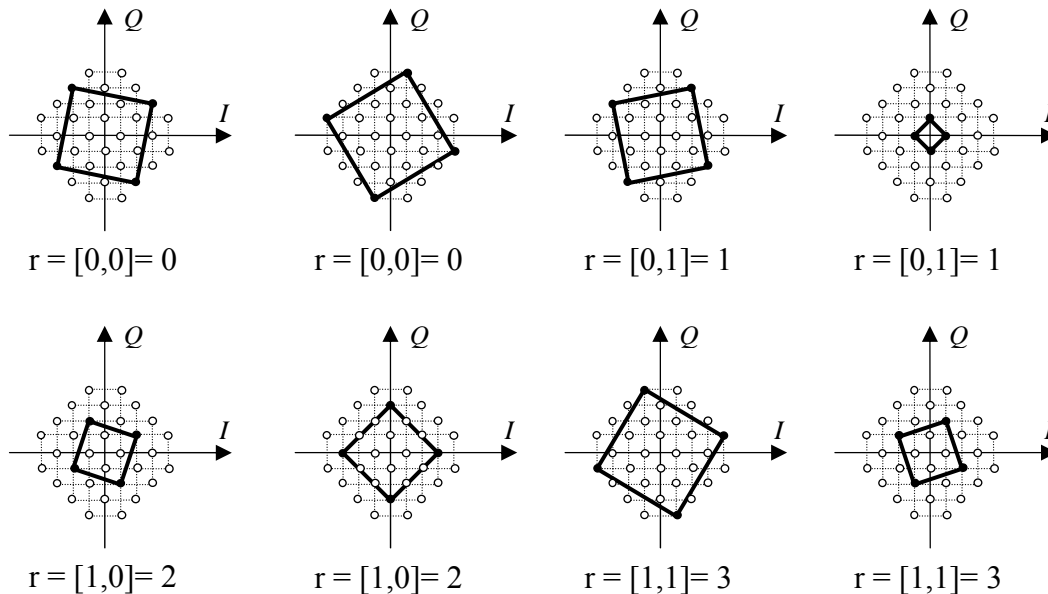
$$u'(n) = \{ u(n) + k \} \bmod 4 \quad (15.12.)$$

amit a 15.10.c.) ábra szemléltet. Ekkor ugyanis a dekódolás:

$$\begin{aligned} q'(n) &= \{ u'(n) - u'(n-1) \} \bmod 4 = \\ &= \{ \{ u(n) + k \} \bmod 4 - \{ u(n-1) + k \} \bmod 4 \} \bmod 4 = \\ &= \{ u(n) + k - u(n-1) - k \} \bmod 4 = \\ &= \{ u(n) - u(n-1) \} \bmod 4 = q(n) \end{aligned} \quad (15.13.)$$

( A **mod 4** műveletekre csak az átvitelek elhagyásának jelzése miatt volt szükségünk, így a bináris alsó két bit helyes, mivel az átvitel nem terjed visszafelé.)

A differenciális kódolásban **részt nem vevő** bitek  $r(n)=[q_4, q_3]$  leképezését ezek után úgy kell elvégezni, hogy azok invariánsak legyenek a forgatással szemben. Ez azt jelenti, hogy egy konstellációs pont  $k \cdot 90^\circ$ -kal elforgatott képeihez ugyan azt az  $r(n)$  értéket kell hozzárendelni. Ezt szemlélteti az alábbi ábra:



15.11. ábra A forgatással szembeni invariancia biztosítása  
(Azok a pontok vannak összekötve, melyeknél az  $r$  értéke azonos)

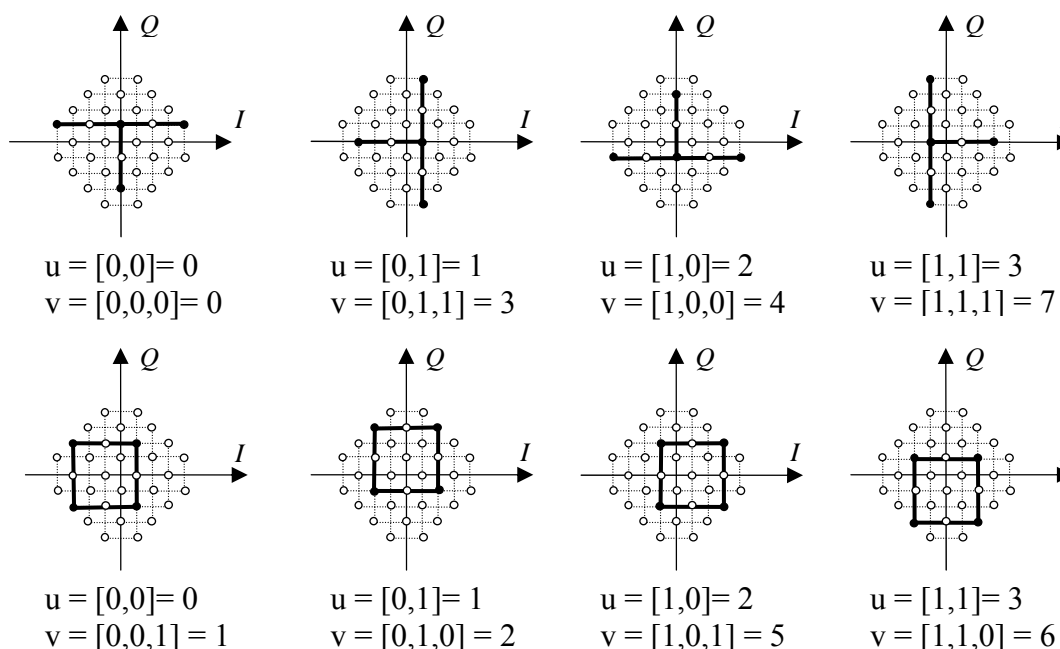
## 2. A trellis-kód moduláció leképzési szempontjai:

A konvolúciós kódoló hárombites  $v(n)=[y_2, y_1, y_0]$  kimenete nyolc különböző értéket vehet fel ( $v = 0,1,2,3,\dots,7$ ). Ezt az értéket tekintjük egy olyan azonosítónak, amelyik a

konstellációs diagram 32 pontos halmazából kijelöl nyolc, egyenként 4 pontot tartalmazó, részhalmazt (*subset*). A részhalmazok kiválasztásánál a következőket kell tekintetbe venni:

- Az egyes részhalmazok pontjai exkluzívak legyenek, vagyis egy pont csak egy részhalmaznak lehet eleme.
- Az egyes részhalmazok pontjai a lehetőleg távol legyenek egymástól. Erre azért van szükség, mert a konvolúciós kódolásban részt nem vevő  $\mathbf{r}(n)=[q_4, q_3]$  bitekre (ami a részhalmazon belüli sorszámot adja) így tudjuk kiterjeszteni a hibavédelmet. A nagyobb távolság ugyanis nagyobb zajt enged meg, anélkül, hogy az adott konstellációs pontra vonatkozó döntésünkben hiba lépne fel.
- A  $\mathbf{v}(n)=[y_2, y_1, y_0]$  részhalmaz azonosító tartalmazza az  $\mathbf{u}(n)=[y_2, y_1]$  bitjeit, amelyekre nézve a differenciális kódolásnál (a 15.12.-ben) egy feltételt tettünk:  $\mathbf{u}'(n)=\{\mathbf{u}(n)+k\} \bmod 4$ . Ez szavakban kifejezve azt jelenti, hogy a részhalmazok pontjait  $k \cdot 90^\circ$ -kal elforgatva az óramutató járásának megfelelően ( $k=0,1,2,3$ ), egy másik részhalmazt kell kapnunk, melyben a megváltozott  $\mathbf{u}'(n)$  szerepel.

Ezen szabályoknak az alábbi ábrában szereplő részhalmazok eleget tesznek:



15.12. ábra A konstellaációs pontok felosztása alcsoportokra  $v = 0,1,2,3,\dots,7$

Az ajánlásban szereplő választással az egyes részhalmazokon belül a pontok távolságát sikerült egységesen azonos távolságban tartani ( $d = 4$  relatív egység) úgy, hogy az egyes *subset*-ek között az elforgatási reláció is fenn áll. Az alcsoporton belül a pontok sorszámának ( $r$ ) természetesen összhangban kell lennie a 15.11. ábra kapcsán elmondottakkal.

### 15.6. A V.17. ajánlás konvolúciós kódolója

A 15.8. ábrán szereplő kódoló egy úgynevezett nemlineáris konvolúciós kódoló. Ez a megoldás az egy bites összeadókon kívül (XOR kapuk) AND kapukat és visszacsatolásokat is

tartalmaz. A hálózat igazságtábláját felírva, abból kiolvashatjuk a már az előző fejezetben bevezetett állapot-átmeneti (**SUS**) és a kimenetet szolgáltató (**VUS**) függvényeket:

$$s(n+1) = \text{SUS}[u(n), s(n)]$$

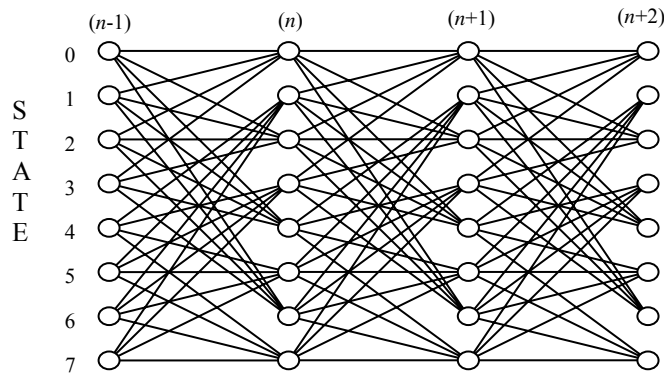
$$v(n) = \text{VUS}[u(n), s(n)]$$

<b>SUS</b>		<b>s</b>							
		0	1	2	3	4	5	6	7
<b>u</b>	0	0	2	4	6	1	3	7	5
	1	2	0	6	4	7	5	1	3
	2	6	4	2	0	5	7	3	1
	3	4	6	0	2	3	1	5	7

<b>VUS</b>		<b>s</b>							
		0	1	2	3	4	5	6	7
<b>u</b>	0	0	0	0	0	1	1	1	1
	1	2	2	2	2	3	3	3	3
	2	4	4	4	4	5	5	5	5
	3	6	6	6	6	7	7	7	7

15.21. Táblázat A V.17. ajánlás **SUS** és a **VUS** függvényei

Az állapot-átmeneti függvény ismeretében megszerkeszthetjük a kódoló *trellis*-ét. A visszacsatolásoknak köszönhetően egy átmenetekben gazdagabb *trellis*-t kapunk. Az egyszerű shift-regiszteres megoldással szemben itt minden állapotból négy másik állapotba juthatunk és minden állapotba négy másik állapotból érkezhethetünk. Az alábbi ábrán három időréshöz tartozóan rajzoltuk le a lehetséges állapot-átmeneteket.



15.13. ábra A V.17. ajánlás *trellis*-e

A dekóder számára szükségünk lesz még az **SPS** és a **VPS** függvényekre, melyeket szintén a kódoló igazságtáblázatából olvashatunk ki. Mivel most az egyes állapotokba négy különböző irányból érkezhethetünk, a  $p$  paraméter ennek megfelelően a  $p=0,1,2,3$  értékeket veheti fel.

$$s_p(n-1) = \text{SPS}[p, s(n)]$$

$$v_p = \text{VPS}[p, s(n)]$$

<b>SPS</b>		<b>s</b>							
		0	1	2	3	4	5	6	7
<b>p</b>	0	0	4	0	4	0	4	0	4
	1	1	5	1	5	1	5	1	5
	2	2	6	2	6	2	6	2	6
	3	3	7	3	7	3	7	3	7

<b>VPS</b>		<b>s</b>							
		0	1	2	3	4	5	6	7
<b>p</b>	0	0	1	2	7	6	5	4	3
	1	2	7	0	1	4	3	6	5
	2	6	3	4	5	0	7	2	1
	3	4	5	6	3	2	1	0	7

15.22. Táblázat A V.17. ajánlás **SPS** és a **VPS** függvényei

### 15.7. A trellis-kód moduláció távolság mértéke

A demoduláció során a dekódolási hiba a csatornában fellépő zaj miatt következhet be. Tételezzük fel, hogy a demodulátorban helyreállított kvadratúra jelkomponenseket additív, egyenként  $\sigma^2$  szórásnégyzetű, Gauss zaj terheli, melynek a spektruma fehér. A zaj a két irányban korrelálatlan. Ekkor az együttes valószínűségi sűrűségfüggvény az  $n$ -ik időrésben:

$$f_n(I(n), Q(n)) = \frac{1}{2\pi\sigma^2} e^{-\frac{[I(n)-I_0(n)]^2 + [Q(n)-Q_0(n)]^2}{2\sigma^2}} \quad (15.14.)$$

Az  $I_0(n)$  és a  $Q_0(n)$  az  $n$ -ik időréshez tartozó névleges ( a zajmentes esethez tartozó) konstellációs pont koordinátái, amelyeket pillanatnyilag még ismeretlen paramétereknek tekintünk. Ha  $N$  számú időrésen keresztül végezzük vizsgálatainkat úgy, hogy az egyes időrészekhez tartozó valószínűségi változókat függetlennek tekintjük, akkor a  $2N$  dimenziós sűrűségfüggvény az alábbi alakban írható fel:

$$f_N(I(1), Q(1), \dots, I(N), Q(N)) = \prod_{n=1}^N f_n(I(n), Q(n)) \quad (15.15.)$$

Ha az  $I_0(n)$  és a  $Q_0(n)$  paramétereket úgy választjuk meg, hogy a fenti együttes sűrűségfüggvényt maximalizáljuk, akkor a helyes döntés sorozat valószínűségét fogjuk maximalizálni (*maximum likelihood method*). Ebben az értelemben járunk el akkor is, ha a (15.15.) logaritmusát vesszük, (mivel az  $\ln(x)$  függvény szigorúan monoton):

$$\ln \left[ \prod_{n=1}^N f_n(I(n), Q(n)) \right] = N \ln \frac{1}{2\pi\sigma^2} - \frac{1}{2\sigma^2} \sum_{n=1}^N \{ [I(n)-I_0(n)]^2 + [Q(n)-Q_0(n)]^2 \} \quad (15.16.)$$

A (15.16.) kifejezést akkor maximalizáljuk, ha a:

$$D = \sum_{n=1}^N \{ [I(n)-I_0(n)]^2 + [Q(n)-Q_0(n)]^2 \} \quad (15.17.)$$

akkumulált euklideszi távolságot (távolság négyzetet) minimalizáljuk (u.i. a legkisebb pozitív számot vonjuk ki egy konstansból).

Tekintsünk most egy  $N$  időrésre kiterjedő kísérletet, amit a valószínűségi folyamat egy realizációjának tekintünk! Legyenek most az  $I(n)$  és a  $Q(n)$  értékek a vett jel kvadratúra komponensei a dekóderben! ( A (15.14.)-ben ezek még a sűrűségfüggvény független változói voltak.) Modellünk értelmében az  $I(n)$  és a  $Q(n)$  értékek az  $I_0(n)$ -nek és a  $Q_0(n)$ -nek az additív zajjal módosított értékei. A célunk az, hogy a vett adatok ismeretében lehető jó becslést adjunk a küldött üzenetre.

Ha az  $N$  időrésen keresztül felrajzolt *trellis*-en az összes lehetséges útvonal mentén kiszámítjuk a (15.17.) szerinti akkumulált távolságot és ezek közül a legkisebbet választjuk, akkor a *maximum likelihood* elv szerinti, (maximális valószínűségű) becslést adunk a vett konstellációs pontokra, azaz a küldött kód sorozatra.

A trellis-kód moduláció Viterbi-dekóderében tehát nem Hamming távolsággal, hanem a helyreállított kvadratúra jel és a névleges konstellációs pontok euklideszi távolságával (távolság négyzetével) fogunk számolni.

### 15.8. A trellis-kód moduláció dekódere (Viterbi dekóder)

A Viterbi-algoritmusról korábban elmondottakat ki kell egészítenünk még néhány gondolattal. A példának választott kódolóban  $\mathbf{r}(n)=[q_4, q_3]$  bitek ugyanis nem vesznek részt a konvolúciós kódolásban, ezért a *trellis*-en történő állapot-átmenetektől az  $\mathbf{r}(n)$  értéke nem függ. Így hiába határozzuk meg a helyes állapot-átmenetet és a hozzá tartozó  $\mathbf{v}(n)$ -et, valamit mondanunk kell  $\mathbf{r}(n)$ -ről is.

Ennek érdekében az algoritmust egy további eljárással egészítjük ki. Ebben az eljárásban feltétel nélkül végigmegyünk a  $\mathbf{v}$ -vel azonosított rész halmazokon és megállapítjuk, hogy az egyes halmazok 4 lehetséges pontja közül (melyeket az  $\mathbf{r}$ -rel azonosítunk) melyik pont van a legközelebb a dekóderben helyreállított kvadratúra jelhez. A bemenő paraméterek: In és Qn, a kimenetek: a **Dist**[v] és az **R**[v] tömbök, amelyek az adott ( $\mathbf{v}$ -vel azonosított) részhalmazban a legkisebb távolságot, ill. az ehhez tartozó  $\mathbf{r}$  azonosítót tartalmazza.

<b>Procedure Get_Distances</b> (In,Qn);	{ Bemenő adatok: In, Qn }
{ for v:= 0 to 7 do	{ Minden csoporton végigmegyünk }
{ Dmin:=Maxint;	{ Dmin legyen a maximális érték }
for r:= 0 to 3 do	{ Minden pontot vizsgálunk }
{ c := 8*r + v;	{ Mi lett volna a küldött kód }
I0 := <b>MAPI</b> [c]; Q0 := <b>MAPQ</b> [c];	{ Az ehhez tartozó koordináták }
D := (In-I0) <sup>2</sup> + (Qn-Q0) <sup>2</sup> ;	{ A. távolság }
If D ≤ Dmin then	{ Minimum keresés }
{ Dmin := D;	{ Az új minimum }
r_min := r ; }	{ A feltételezett kódot elmentjük }
};	{ end of for r }
<b>Dist</b> [v] := Dmin;	{ Elmentjük a v-hez a Dmin-t }
};	
<b>R</b> [v] := r_min;	{ Elmentjük a v-hez. az rmin-t }
};	{ end of for v }
};	{ end of procedure }

A túlélő utak keresése hasonló a korábban elmondottakhoz. Két egymásba ágyazott ciklusból áll, a külsőben feltétel nélkül végigmegyünk az összes állapoton ( $\mathbf{s}$ ), míg a belsőben a lehetséges beérkezési irányokon megyünk végig ( $\mathbf{p}$ ). A lehetséges előző állapotok ( $\mathbf{se}$ ) ill. az ezen állapot-átmenethez tartozó ( $\mathbf{v}$ ) ismeretében frissítjük az akkumulált távolságot.

<b>Procedure Survival_Path</b> ;	{ A túlélő utak keresése }
{ for s:= 0 to 7 do	{ Minden állapoton végigmegyünk }
{ Dmin:=Maxint;	{ Dmin legyen a maximális érték }
for p:= 0 to 3 do	{ Minden befutó ágat vizsgálunk }
{ se:= <b>SPS</b> [p,s];	{ Mi lett volna az előző állapot }
v := <b>VPS</b> [p,s];	{ Mi lett volna a v értéke }
D:= <b>Dist</b> [v] + <b>ACC_Dist</b> [se];	{ Az átmenethez tartozó távolság }
If D < Dmin then	{ Minimum keresés }
{ Dmin := D; pmin := p; vmin:=v ;	{ end of if }
};	{ end of for p }
<b>Temp_Dist</b> [s] := Dmin;	{ Ideiglenesen elmentjük az új a.t. }
<b>State_From</b> [s,0]:= <b>SPS</b> [pmin,s];	{ Mi lett volna az előző állapot }
rmin:= <b>R</b> [vmin];	{ Mi lett volna az r }
<b>C</b> [s,0] := 8*rmin + vmin;	{ A feltételezett kódot elmentjük }
};	{ end of for s }
for s := 0 to 7 do { <b>Acc_Dist</b> [s] := <b>Temp_Dist</b> [s] ;	{ Az új akkum. távolság frissítése }
};	{ end of procedure }



Az eljárás kimeneti változói hasonlóan a korábbiakhoz az **Acc\_Dist[s]** és a **State\_From[s,0]** tömbök. A korábbiaktól eltérően eltároljuk a **C[s,0]** kétdimenziós tömb 0-ik oszlopában az állapotokhoz tartozó legvalószínűbb **c** értékeket (ami tartalmazza a **v** és **r** értékeket is).

A nyomkövető eljárás teljesen hasonló a korábban elmondottakhoz. Ebbe az eljárásba már beleraktuk az **Acc\_Dist** túlsordulását elkerülő módosítást is (nevezetesen azt, hogy a minimumot mindegyik értékből levonjuk). A késleltetési értéket  $L=15$ -re választottuk. Az algoritmus kimeneti változója a **c\_out** érték.

Prucedure <b>Trace_Back</b> ;	{Nyomkövetés	}
{ Dmin := Maxint;	{ Maximális érték	}
for s := 0 to 7 do	{ A minimum megkeresése	}
{ if <b>Acc_Dist[s]</b> < Dmin then	{ Minimum keresés	}
{ Dmin := <b>Acc_Dist[s]</b> ; smin := s }	{ smin értéket kap	}
};	{end of for s	}
for s := 0 to 7 do { <b>Acc_Dist[s]</b> := <b>Acc_Dist[s]</b> - Dmin };	{A túlsordulás elkerülése	}
snext:=smin;	{A kezdő állapot :smin	}
for n:= 0 to 14 do { snext := <b>State_From</b> ( snext,n); }	{Vissza az utolsó előtti időig	}
c_out:= <b>C</b> [snext,15];	{Ez mutat a küldött kódra a C-ben	}
};	{ end of procedure	}

A késleltetési eljárásnak most a **C** tömbre is ki kell terjednie. Cirkuláris címezéssel ez az eljárás elmaradhat a valós idejű megvalósításban.

Procedure <b>Delay</b> ;	{Cirkuláris buffer esetén nem kell }
{ for s := 0 to 7 do	{Minden állapotra }
{ for n :=15 downto 1 do	{A multból a jelen felé haladva }
{ <b>State_From</b> [s,n] := <b>State_From</b> [s,(n-1)];	{State_From öregbítése }
<b>C</b> [s,n] := <b>C</b> [s,(n-1)];	{ C öregbítése }
};	{end of for n }
}	{end of for s }
};	

### 15.9. A trellis-kód moduláció nyeresége

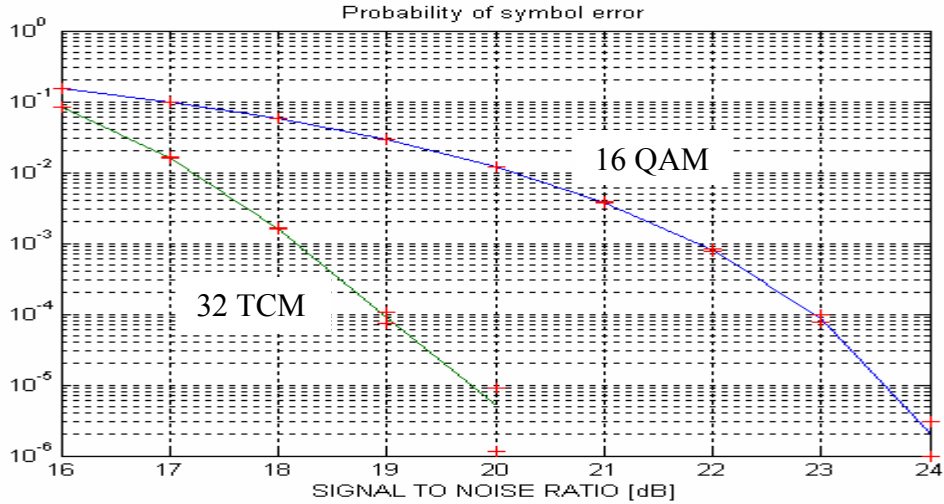
A trellis-kód moduláció használatával elérhető nyereség bemutatására készítettünk két DSP programot. Mindkettő a V.17. ajánlásban szereplő modulációs eljárások egyikét használja, az első program a 32 QAM-et alkalmazó TCM eljárást, a másik egy konvolúciós kódolás nélküli 16 QAM-et. Így azonos sáv szélesség mellett azonos effektív adatátviteli sebességeket tudunk összehasonlítani.. (Azonos szimbólum sebesség esetén a szimbólumonkénti effektív bitek száma azonos. )

Mindkét program hasonló felépítésű volt: tartalmazta a modulátort és a demodulátort is, hogy elkerüljük az időzítés visszaállításból adódó hibákat.

A modulátor bemenetére korrelálatlan bitsorozatot adtunk. Az ezekből a bitekből képzett szimbólumok száma egy mérés során 1 millió volt.

A modulátor I,Q kimeneteihez állítható nagyságú, független fehér Gauss zajt adtunk. A demodulálás után számoltuk az egyes szimbólumok átviteli hibáit. Egy kiválasztott zajszinthez tartozó mérést 20-szor ismételtünk meg, melynek hiba számait átlagoltuk és ezt tekintettük az adott beállításhoz tartozó hiba valószínűségnek.

A kapott eredményeket a 15.14. ábrában tüntettük fel. A mérés szórását a folytonos vonalak mellett látható '+' jelek mutatják. ( A szórás természetesen a  $10^{-6}$  hibavalószínűség környezetében relatíve nagy, mivel a sorozatok csak  $10^6$  szimbólum hosszúságúak voltak)



15.14. ábra A hiba valószínűségének alakulása a jel-zaj viszony függvényében

Az ábrából láthatóan például  $10^{-4}$  hibavalószínűség eléréséhez kb 19 dB jel-zaj viszony elegendő a 32 TCM esetben, míg a 16 QAM-nél ehhez már kb 23 dB kell. E két érték különbsége a TCM eljárás nyeresége.

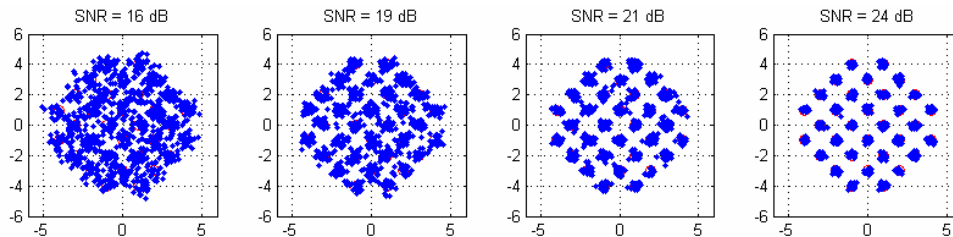
A jel-zaj viszony számításához még egy megjegyzés:  
A jel teljesítményének a 32 TCM esetében:

$$P_{jel} = \frac{1}{32} \sum_{c=0}^{31} [I_0^2(c) + Q_0^2(c)] \quad (15.17.)$$

mennyiséget, míg a zaj teljesítménynek a:  $P_{zaj} = 2\sigma^2$  értéket tekintettük, ahol  $\sigma$  a kvadratúra irányokban hozzákevert, független Gauss zajok szórása. (A 16 QAM esetében hasonlóan jártunk el.) A jel-zaj viszony:

$$SNR^{dB} = 10 \lg \frac{P_{jel}}{P_{zaj}} \quad (15.18.)$$

Végezetül az alábbi ábrán egy MATLAB szimuláció eredményeként született konstellációs diagramokat mutatunk meg, amin az látható, hogy hogyan változik a jel minősége a jel-zaj viszony függvényében :



15.15. ábra A konstelláció a jel-zaj viszony függvényében  
(a pontok száma 1000)