LONG COLUMN-CODED REGULAR KNOT
(MADE ON A THK SHADOW)

A **“LONG”** knot is one were **L – B >=2**.
In this document (**L – B**) is “very much” superior to “ 2”.

“very much” superior as in
101 L 8B (L-B = **93**)
or **101** L 7B (L-B = **94**)
or **137** L 8 B (L-B = **129**) from **Ed PASS** from Arizona
or **135** L 8 B (L-B = **127**) from **Jim BARCUS** from Arizona .

I will say from a “practical point of view : it is “l—o__o—o__n—g” when (L –1) and B are such that :

you cannot write enough ‘B long’ Bight sequences to cover the (L-1) crossings using the longest dimension of a sheet of paper !

See here : **101 L 7 B** that mean **100 crossings** and a **Bight sequence of 7** so **100 / 7 ! 14 full sequences and a part of a 15th**.

Just imagine

IP (Integer part) FP (Fractional Part)

IP (101 / 7) = 14 (remainder = 3) IP (101 / 8) = 12 (remainder = 5)
IP (137 / 8) = 17 (remainder = 1) IP (135 / 8) = 16 (remainder = 7)

This means that you will need in order to cover the **L – 1** crossings in a H-P to align 12 + 1 to 17 + 1 Bight *sequences* to make the Bight *Algorithm*.

A Bight algorithm that would be for each Half-Period (H-P) 100 to 136 digits long to write and to read without one mistake made!

Better you than me !

Once again SCHAAKE and TURNER provided an easy way out.

I will personally summarise the way out as : If you cannot go “SERIAL” then go “ PARALLEL”.
So to speak.

In other words just STACK one upon the other enough ‘B long’ Bight sequences to “cover up” the equivalent of the one long line Bight Algorithm.

Some simple math to put back in mind:

L / B = IP + (FP * B) = IP + remainder

or
L = (B * IP) + remainder
L = (B * n) + remainder IP = n

(L) mod B
(-L) mod B or (B – L) mod B

now with 17 L 7 B for the exemplification
17/7 = 2 as IP + 3 as remainder (FP = 0.4285 so 0.4285 * 7 = 3)
17 = (7 * 2) + 3
17 = (7 * 2) + 3
(17) mod 7 = 3
(-17) mod 7 = 4 (7 – 17) mod 7 = (-10)mod 7 = 4

Suppose we are "doing" the **101 L 7 B** LOOOOONG standard knot :
(I am still with Schaake mandrel frame)
We need a Bight sequences (complementary and periodic) that will be 'B long' and numbered not from 1 to B but from 0 (to B-1) so from 0 to 6.

To "cover" for the (L-1) crossings in a H-P we will need ((L-1) / B) rows (rounded to the superior integer.)

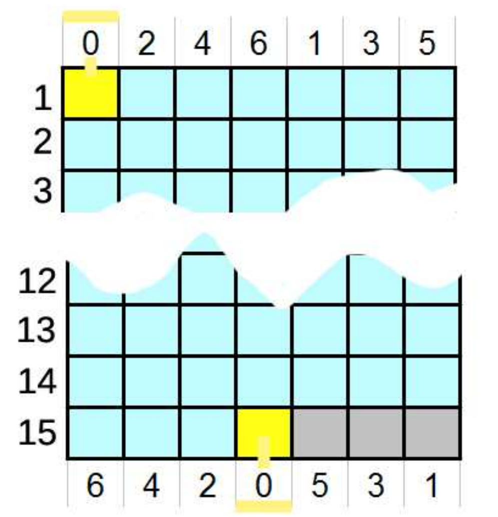
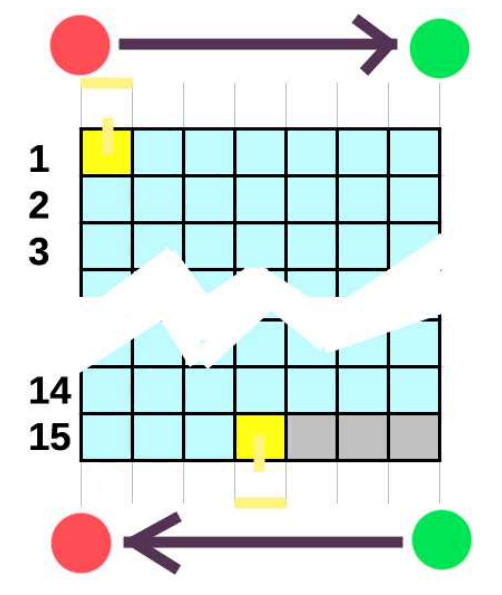


Fig 1



That is IP (integer part) of the operation plus one (or as some calculators do **CEIL**ing of ((L-1) / B).)

101 – 1 = 100 100 crossings in a H-P

100 / 7 = 14.2857 so **CEIL**ing is 15

in fact 15 is also **N** or IP plus one
14 + 1 = 15

verify 14 * 7 = 98

We are missing 2 crossings so we must put in a 15th row of which not all the cases / cells will be used.

At the bottom and at the top of each B columns there will be a digit as *Fig 2* we write the **COMPLEMENTARY** Bight *sequence* in the Row immediately above ROW 1 and the **PERIODIC** Bight *sequence* in the Row immediately under the 15th ROW.

ALWAYS TAKE CARE TO remember the **direction of writing/ reading** for the **COMPLEMENTARY** and **PERIODIC** Bight *sequence*.

Remember that it is very unwise to suppress the "0" on the leftmost side of the **COMPLEMENTARY** Bight *sequence* and the rightmost side of the **PERIODIC** Bight *sequence*.

These "0" not to be dispensed with are 'place holder' for the BIGHT RIM (bight boundary for S & T) (left and right rim in the horizontally held mandrel reference, bottom and top rim in the vertically held cylinder reference).

As the Rows 1 to 15 are to stand for the long Bight *Algorithm* we need to figure the "0" somehow in the CODE CELLS.

So the first left Cell of the first Row (topmost == ROW numbered 1) will be painted yellow (in here – you can also just put a cross in it) just so that it is not actively used but still kept in mind and calculation).

Then we count 100 cells (the blue ones – easy, 100 is one row of (B-1) here 6 cells, plus 13 row of 7 cells = 91. That gives us 91 + 6 = 97 ; we are still missing 3 to be at 100 full complement.

So we count 3 cells in the next and last (15th) ROW and the next cell is painted yellow to stand for the right side bight rim.

In fact there is a quicker way :

- paint in yellow the first cell in ROW 1
- use N (the IP of L/B) to count the ROW 1 to 14
- in the 15th row you use the 'remainder" of L/B ,(here 3)
-

that way you get immediately your (L-1) cells for as much crossings and the RIM are 'flagged'.

Now we are at the stage shown in *Fig 2*

We still missing some elements :

--- The coding of crossings in the first H-P as seen by the SPart-WEnd directional arrow (vector) in the finished knot.

--- for ODD LEAD knots (so EVEN number of CROSSING) we need to "mark" somehow the length wise middle of the mandrel (the 'equator' of the mandrel)

(L-1) / 2 EVEN / 2 imply half and half separation here (101-1) / 2 = 100/2 = 50

50 crossings before and 50 after the dividing mark (in Fig 3 it is the white bar with a red dot) .

You can count the cells one by one.

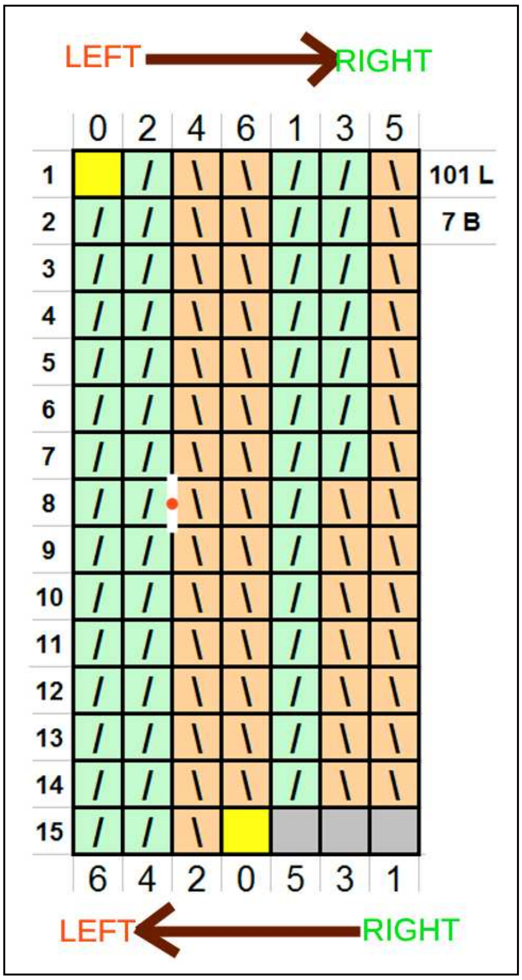
Or you can set yourself to “brain engaged” mode and use mathematics. (easy ones)
50 – 6 (the ROW 1 blue cells) = 44
44 / B = 44 / 7 = IP of 6 and remainder of 2 so after the ROW 1 you count 6 rows, you are at ROW 7.
You go on ROW 8 and simply count the “remainder” : 2 : mark is between cell 2 and cell 3

THERE IS A FASTER METHOD

(L-1) / 2 is the number of cells before and after the mid-mark in an **ODD LEAD** knot.

Mark will be (that is rather evident) in cell ((L-1)/2) + 1) we have to add 1 to compensate for the yellow cell and be certain to be above the first 50 and under the second 50 so to speak .

Separating mark will be between cell ((L-1)/2) + 1) and the adjacent one in the left tp right direction, cell ((L-1)/2) + 1 + 1)



That can be written respectively
((L-1)/2) + 2/2 == (L – 1 + 2) / 2 ==(L + 1) / 2
and ((L-1)/2) + 2/2 + 2/2 == (L – 1 + 2 + 2) / 2 ==(L + 3) / 2

taking again (((L-1)/2) + 1) and making C = (L-1)/2

We are going to use a **MODULO** to render “transparent” whole rows where the separating mark cannot be situated.
(L+1) / 2) mod B == ((101 + 1) / 2)mod 7 == (102/2)mod 7 == (51)mod7 == 2

2 == second cell of 8th row

(51 = (7 * 7) =+2)
(L + 3) / 2)mod B == ((101 + 3) / 2)mod7 == (104/2)mod 7 = 52 mod 7 = 3
separation is between CELL 2 and CELL 3 of ROW 8

8 because 51 = 7 * B + 2 == 7 * 7 + 2
52= 7 * B + 3=7*7 + 3

FULL rows plus a bit of the 8th to “use” the remainder on so :

Formula are
((L+1)/2) mod B and ((L+3)/2) mod B
ROW is IP of (L-1)/(2*B) plus one
here (101 –1 / 2 * 7) + 1 == (100 / 14) + 1 == IP is 7 plus 1 == 8

When 'B' is EVEN then exist a symmetry relative to the lengthwise middle, this imply that if you want a different pattern for the 2 bight rim then you have to chose 'B' ODD.

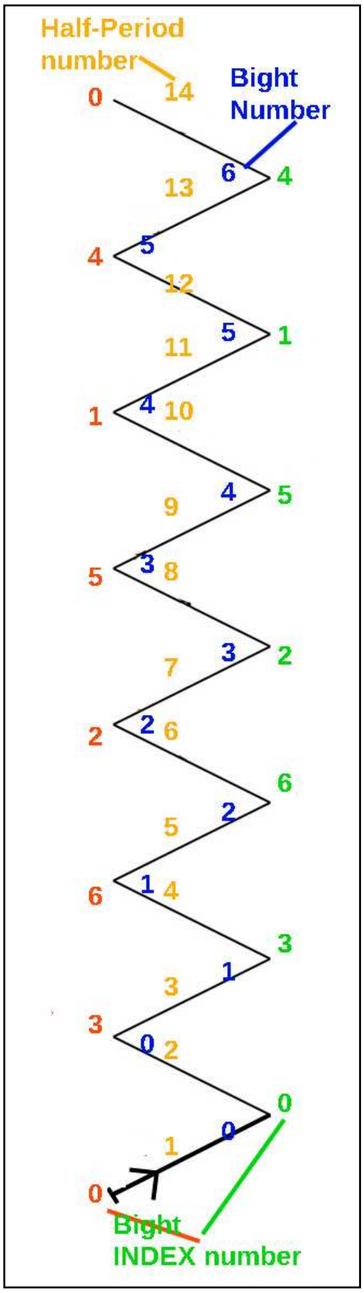
We are now ready (almost) to mark the crossings code on the first H-P as seen by the Wend in the finished knot.

We need to calculate
DELTA* (I suggest you use the adequate program of EMU48 if you want that !)

((L)mod B = (101) mod 7 = 3

(-L)mod B = (-101) mod 7 = 4
or easier as you don't risk forgetting the minus sign before L use (B – L)mod B == (7 –101)mod 7 ==(-94)mod 7 == 4

Use (-L)mod B = (-101) mod 7 = 4 as "stepping" to write the **COMPLEMENTARY** (of course it is more direct with DELTA* which is 3 so you write directly 0 3 then 0 3 6 then –modulo 7 – 0 3 6 2 and so on)



0 x x x **1** x x **0 2** x x **1** x x

0 2 x **1 3** x x **0 2 4** x **1 3** x

0 2 4 x **1 3 5** **0 2 4 6 1 3 5**

0 2 4 6 1 3 5

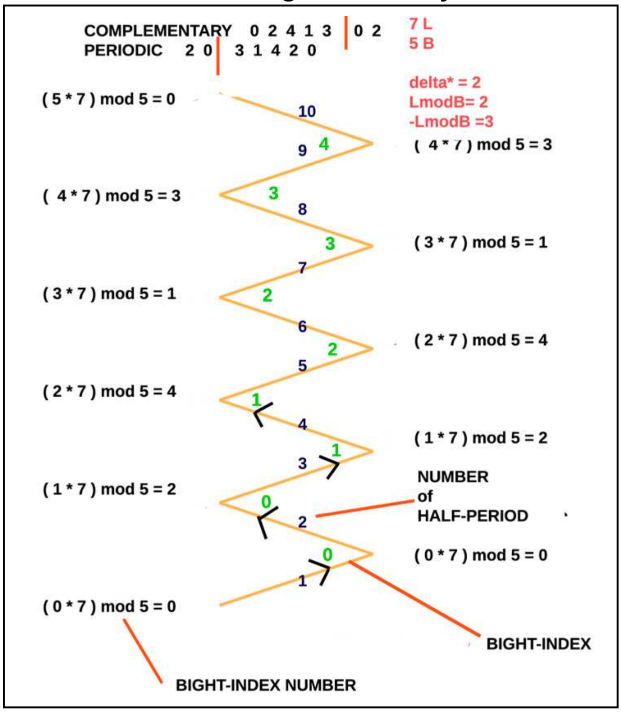
JUST REVERSE to get the
PERIODIC
5 3 1 6 4 2 0

The writing of the **COMPLEMENTARY** is easy : usual direction of reading and writing **LEFT** to **RIGHT** , one digit above each column

The writing of the **PERIODIC** is a slightly more difficult affair : **STARTING CASE with the ‘0’ will be the one just under the yellow cell figuring the right side Bight Rim.**
Then you write “Arabic fashion”; that is **RIGHT** to **LEFT** what you read **RIGHT** TO **LEFT** in the PERIODIC and going “circular” so after entering **6 4 2 0** you are left with **1 3 5** to write and 3 empty cells on the rightmost part so in the end you will have written
6 4 2 0 5 3 1

This illustration is from another knot (the one use to write the USER’S TIPS for BIHP).
It shows all you need though.

(you will do good to read **BALG** and **BIHP** user’s tips to grasp more easily what is said here – I cannot spend my time always repeating the same things, you need to make you own efforts and READ SCHAAKE’s *THE BRAIDER*)



Now for an example

0 2 4 6 1 3 5
here in the middle is the stack of coding rows
6 4 2 0 5 3 1

This works just as the other BIGHT ALGORITHM for the THK

ODD numbered Half-Period (going from low left to up right) will use
0 2 4 6 1 3 5 (read left to right)
then you work your way in the stack of row as if normally reading Row (1) is read **left to right**
then Row (n + 1) is read **left to right**
then Row (n + 1 + 1) is **left to right** …..

EVEN numbered Half-Period (going from low right to up left) will use
6 4 2 0 5 3 1 (read right to left)
then you work your way in the stack of row in a special manner
read Row (n) from **right to left**
then Row (n – 1) is read **right to left**
then Row (n – 1 – 1) that is Row (n – 2) is read **right to left** then….

Say we are out to follow (when doing the knot) the coding for **Half-period N°3**
Then as this is an **ODD** numbered period we go on the upper bight sequence

0 2 4 6 1 3 5

We read on the before last illustration given that the **H-P 3 ODD** numbered (low **left** to up **right**) half-period that starts from a bight that has a green **0** as bight number or we then calculate (i) as in **i = (H-POdd – 3) / 2**
(3 – 3) / 2 = 0
and we make use of the columns that are equal or less than (i)
0 2 4 6 1 3 5
read **left** to **right** and **downwards** this gives **OVER 14** **O 14** [**/**]

Now for **H-P 5** , again an **ODD** numbered (low **left** to up **right**) half-period
i = (5 – 3) / 2 = 1
0 2 4 6 1 3 5
read **left** to **right** and **downwards** this gives **O 28** [**/**]

now for **H-P 6** , that is and **EVEN** numbered (low **right** to up **left**) half-period
i = (H-PEven – 2) / 2
i = (6 – 2) / 2 = 2
we go on the lower Bight sequence
6 4 2 0 5 3 1
and read what is equal or less than(i)
6 4 2 0 5 3 1
Read **right** to **left** and **upwards**

That will give **O 43** [****]

Now for **H-P N° 12**, that is an **EVEN** numbered (low **right** to up **left**) H-P so we go on the lower **6 4 2 0 5 3 1**I = (12 – 2) / 2 = 5

So we read **right** to **left** and **upwards** all that is equal or less than 5
6 4 2 0 5 3 1
we are making use of 6 columns (4 with 14 codes and 2 with 15 codes which makes 4*14 + 2*15 = 86 codes all told)

this is what you will be reading on the rulers :
O U O OU O O UO OU O O UO OU O O UO OU O O UO OU O O
UO OU O O UO OU O O U O U U O O U O U UO O UO U U O O
U O U UO O U O U U O O U O U UO O UU O U U O O U

Over 1 [****] - **Under 1** [**/**]– { **O 2 – U 1** }[**\\ \\ /**] 14 times – { **O 1- U 2- O 2 -U 1**} **[\\ / / \\ /**]] 7 times
1+1+(2+1) * 14 + (1+2+2+1) * 7 == 2 + (3*14)+(6*7)=2 + 42 + 42 = 86 codes

You see that it is easy to verify that you have not forgotten some ‘en route’

Now for **H-P 11** that is and **ODD** numbered (low **left** to up **right**) H-P so we go to the upper

0 2 4 6 1 3 5

i = (11 – 3) / 2 = 4

So we read **left** to **right** and downwards all that is equal or less than 4

0 2 4 6 1 3 5

this is

Over 1 [**/**] - **Under 1** [****]– { **O 4 – U 1** } [**/ / / / **]7 times – { **O 1- U 1- O 2 -U 1**}[**/ \ / / **] 7 times

DON’T WRITE THE CODE JUST READ IT !

Note : **** means **UNDER** for an **ODD** H-P but **OVER** for and **EVEN** H-P
/ means **OVER** for an **ODD** H-P but **UNDER** for and **EVEN** H-P

If you use (for a calculator program that you want ‘general’)

0 for Under and **1** for Over

then you will have to “translate”

a sequence written for **ODD POINT OF VIEW** **0 0 1 0 1 1**

into **EVEN PERSPECTIVE** / **1 1 0 1 0 0**

Which is really easy to do :

Add 1 to each digit in the sequence to be translate

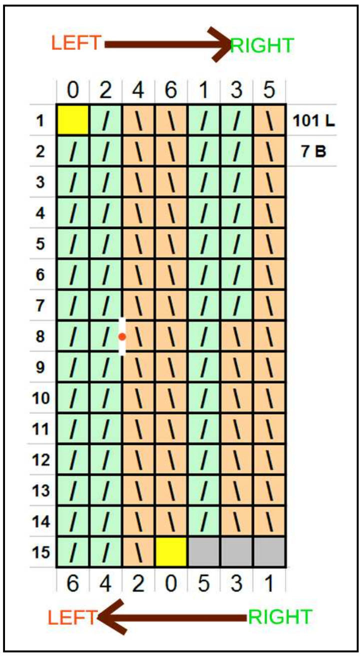
0 0 1 0 1 1

1 1 2 1 2 2

apply MODULO **2** and you get

1 1 0 1 0 0

the translation is finished. Much easier for computing that using **/** low left to up right slant) and **** (right low to up left slant) characters, the code of which you have anyway to put in digits !



Even for a human it is easier IMO to use O and 1 instead of these very confusing signs **/ ** compare to the speed of visually interpreting **0 1**

There is an awful lot less resemblance between **0 1** than between **/ **

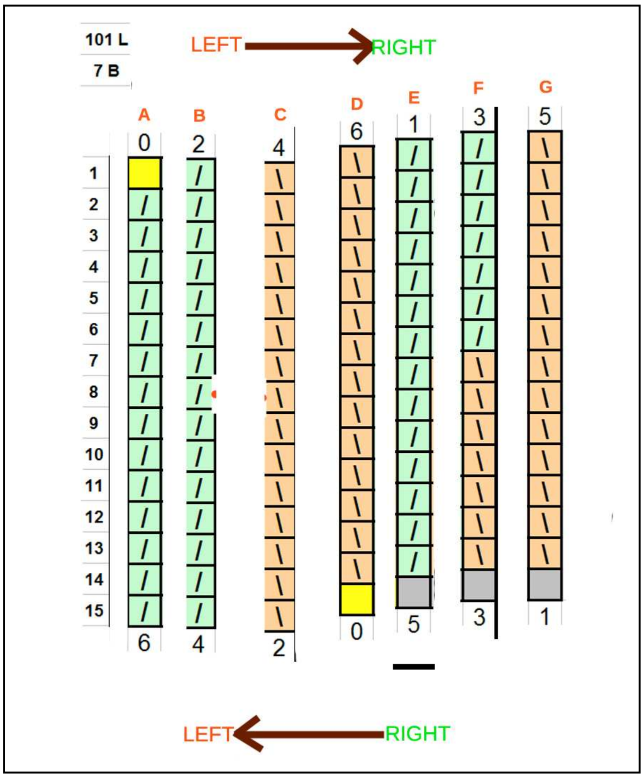
/ \ are IN FACT used here as IDEOGRAMS and not the easiest to distinguish one from the other !

Now for a slight modification of mine that I believe make the reading easier and serve as a “materialised” disambiguation of the H-P your are reading :

Instead of using a sheet of paper I proposed to do a “ sort of Napier’s bones” set of PVC rules :

One rule for the column used to make the numbering of the rows.

As much rules as there are BIGHT in the knot (Of course don't try a 254 L 37 B long column coded knot !)



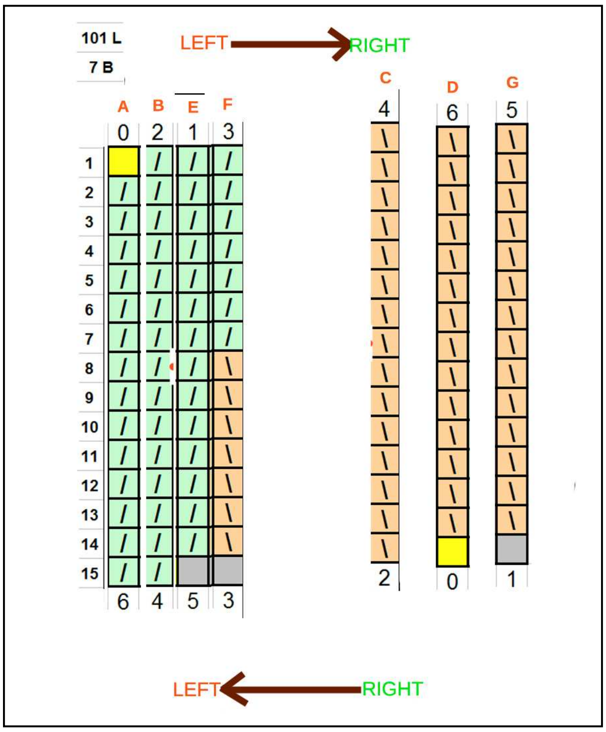
Now we have 8 rules of flat PVC strips.

The first on the right is inscribed (permanently) with the Row Numbering

The 7 (in this particular case) other rules receive a a capital letter each before anything they will get latter on (in permanent making and RED or something truly "flash")

This letter is set above the line that will serve to write the BIGHT ALGORITHM, in alphabetical order (this is what will allow you to always get back to "situation zero")

Dispose your rules in alphabetical order.
Write the ALGORITHM and the CODING.(not with permanent ink !)
Now say you have to "read" the coding for **H-P 5**



You calculate (ODD H-P so low **left** to up **right**)) the (i) value
I = (H-Podd – 3)/2 == (5 – 3) / 2 = 1

Read on the UPPER sequence all that is equal or less than 1, the calculated (i) value.

With a fixed paper diagram you get

A B C D E F G
0 2 4 6 1 3 5

but you need to read only

A B C D E F G
0 2 4 6 1 3 5

You will find cumbersome and prone to make you dizzy and mistaken the

	B	C	D		F	G
	2	4	6		3	5

with the 'PVC rules' you just put

A	E
0	1

alongside each other (discarding the others for the moment) and read.

ALWAYS VERIFY that from LEFT to RIGHT the RULES ARE IN ALPHABETICAL ORDER

That is what maintain the original order in the BIGHT SEQUENCES (Up and Bottom) and do not put the BIGHT ALGORITHM out of kilter !

Suppose now that it is **H-P 10** that you want (i = (H-Peven – 2) / 2 == (10 – 2)/2 =4 **EVEN** so you read the bottom sequence (**right** to **left** remember ?)

Instead of having to deal with a full fixed complement :

A B C D E F G
0 2 4 6 1 3 5
rows here in the middle
6 4 2 0 5 3 1

that makes it a tailored :

B C D F G *(alphabetical order must be complied with **ABSOLUTELY**)*
2 4 6 3 5
rows here in the middle
4 2 0 3 1

That's all folks !

PS : of course and as usual **the coding sequence to 'enter' is the one seen in the FINISHED knot by the FIRST HALF-PERIOD**, but of course too no one had forgotten that. Nor have you forgotten that the number of half-turn (180° - Pi radian) is given by L / B so the number of full 360° turns is given by L / 2*B

A farewell quote from SCHAAKE that I find quite in accord to my past and present thinking :

"We have remarked earlier that braiding a knot by means of its weaving-pattern is not a recommendable method, for the reasons quoted. However, there will always be braiders who have no desire to produce original or varied work, some of whom will swear by the "weaving-pattern" method of braiding. Needless to say, these braiders will never make the grade of braiding-artisan."

Weaving-braiding : braiding by laying a H-P that is immediately adjacent to the preceding one (of same orientation odd with odd ; even with even). That can happen ONLY if $L_{mod\ B} = 1$ or if $L_{mod\ B} = B - 1$