BEAUTIFUL UNIVERSE SIMULATION
Part 1: proof of concept of energy transport in a simple 2-D array of nodes

Vladimir F. Tamari

vladimirtamari(at)hotmail.com Tokyo, July 16, 2016

Abstract

The *Beautiful Universe* [1] model is a universal cellular automata of rotating dielectric dipolar nodes exchanging angular momentum with neighboring nodes to make radiation, matter, and space itself. In this paper a simplified simulation of non-rotating nodes that nevertheless obey the momentum-exchange 'rules' of the model provide a qualitative picture of energy transport in the model and can be considered a preliminary proof of concept on which to build further work

I- Introduction

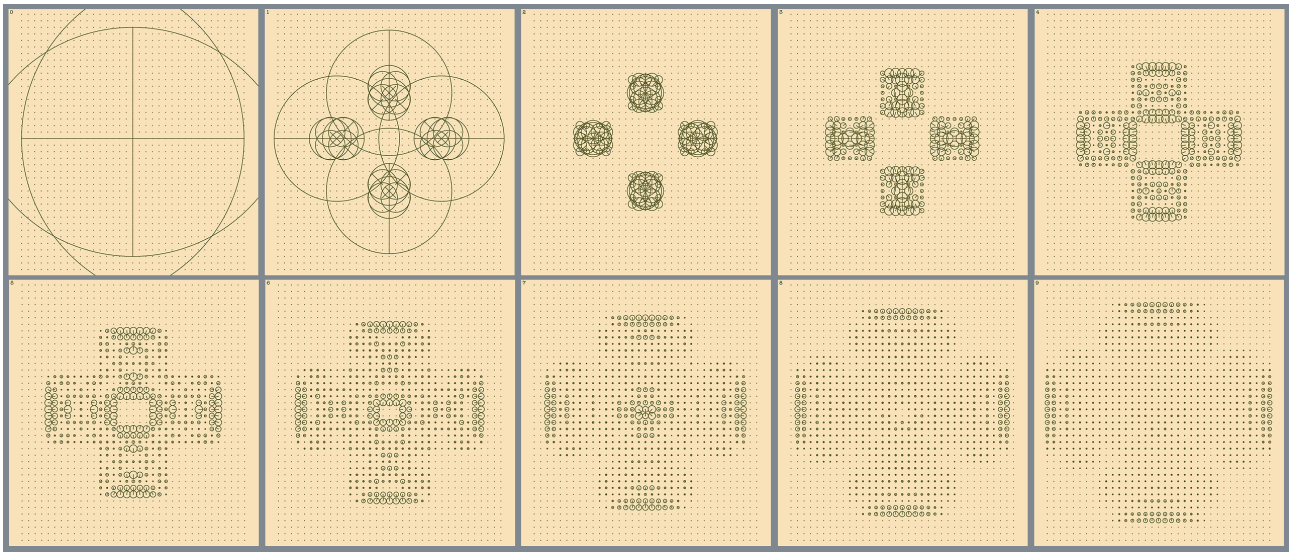


Fig. 1 The first ten frames of the 2D simulation of four nodes of the initial state (top left) spread to adjoining nodes and spread then overlap. The size of a circle centered on a node indicates its angular momentum, while its radius shows the direction the energy is directed. As the energy spreads to more and more adjoining nodes each the a node carries less and less energy, but the total momentum is conserved.

1 Tamari, Vladimir F., *Beautiful Universe: Towards Reconstructing Physics From New First Principles* (self-published online 2005) http://vladimirtamari.com/beautiful_univ_rev_oct_2011.pdf (Web access: July 6, 2016)

Beautiful Universe:Towards Reconstructing Physics From New First Principles (BU) the rudimentary 2005 model, generalizes a1993 result that a [2]single dipole provides a unified electromagnetic, quantum and relativistic field, including gravity. BU proposes that the Universe (space, matter, radiation, energy, dark matter and energy etc.) is made up of a 3D Cellular Automata (CA) matrix made up of just such dipolar nodes rotating in place and exchanging angular momentum in units of Planck's constant (h). In 2014 Gerard 't Hooft showed that Quantum Mechanics can be modeled by $(CA)^3$, which encourages the (BU) approach.

(Fig. 1) shows a preliminary proof-of-concept of how nodes transfer angular momentum to their neighbors according to a specific rule depending on the energy of the node being considered and inversely on the energy of its neighbor to which momentum is transferred. The rule is explained in Section II, and is is contrasted to the ordinary understanding of how (CA) cells interact merely by turning themselves and/or their neighbors on and off, for example in Stephen Wolfram's NKS[4], which speculated that aspects of physics can be modeled as (CA).

While this simulation is limited in the several ways discussed in Section III. Its preliminary success however paves the way for more realistic simulations of (BU) in a 3D array whereby each node interacts with its neighbor with +-, -+, poles attracting and ++, - - repelling each other, in an attempt to model energy-particle conversion according to e=mc^2, gravity, Lorentz Transformations, quantum mechanical effects such as uncertainty, dark matter and energy and so forth as detailed in the (BU) proposal.

II -The simulation of wave transport from node to node

In a two dimensional orthogonal array of nodes forming r rows and c columns the momentum vector of a node is m(r,c). Transfer from a given node (a) to its neighboring node (b) (the recipient) located next to it in the same row is:

$$dmab(r,c)= ma(r,c)/v(1+|mb(r+1,c+1)|) \quad (1)$$

where ma and the mb are the momentum components of the nodes in question along the line ab joining them. The absolute value of mb is used indicated by the || bars. dmab < ma is the is the portion of node (a)'s momentum transferred to (b) in unit time 'tick'. The momentum transferred from (b) to (a) is similarly calculated:

$$dmba(r,c)= mb(r,c)/v(1+|ma(r+1,c+1)|) \quad (2)$$

and unless mb=ma, dmab is usually unequal to dmba. The 2D simulation shows energy spreading as expected. In the resulting graphic The simulation scheme involves the following steps for each time tick (s):
1- Define initial momentum m(r,c) and 'phase' t(r,c) i.e. angle of the momentum of each

2 Tamari, Vladimir F., *United Dipole Field*, 1993, published online 2008 http://arxiv.org/pdf/physics/0303082 (Web access: July 6, 2016)

3 Hooft, Gerard, *The Cellular Automaton Interpretation of Quantum Mechanics* https://arxiv.org/abs/1405.1548 (Web access 8 July, 2016)

4 Wolfram, Stephen, A New Kind of Science , (2002) http://www.wolframscience.com/nksonline/toc.html (Web access: July 6, 2016)

node. Add up the total energy of all the nodes (mom) as a scalar.

2- Display the data. The radius of the circle surrounding a node indicates its energy, while its direction is shown by the angle made by the radius shown. The data could also be displayed as a vector field.

3- For each time-tick s = 0, 1,2, … and for each node in turn on row r= 1,2,3... and column c= 1,2,3, …, and using the convenient scanning scheme of assigning names (a), (b), (c), (d) to the four contiguous nodes to the left and below a given node (n) as described in the in Section IV, Use Eq(1) and Eq.(2) to systematically calculate the energy transfer between all unique pairs of contiguous nodes in the array. Each node has eight contiguous nodes surrounding it, except at the edges of the matrix.

4- Calculate the (x) and (y) components of the resulting transferred momenta, and summate them for each node. This is works because the momentum is added and subtracted linearly.

5- Add up the total resulting total momentum (momtot) of the array and, to conserve momentum, multiply each node's new momentum by a renormalization scalar

ren = mom /momtot (3)

This may seem like an artificial step casting doubt that (BU) is true to the workings of nature, where momentum is conserved in due course. However, due to the step-by-step scheme of transfer between a node and its neighbors, rather than an all-at-once determination of all transfers, a spurious numerical imbalance is introduced for example when a node may end up contributing a total of more energy than it had initially.

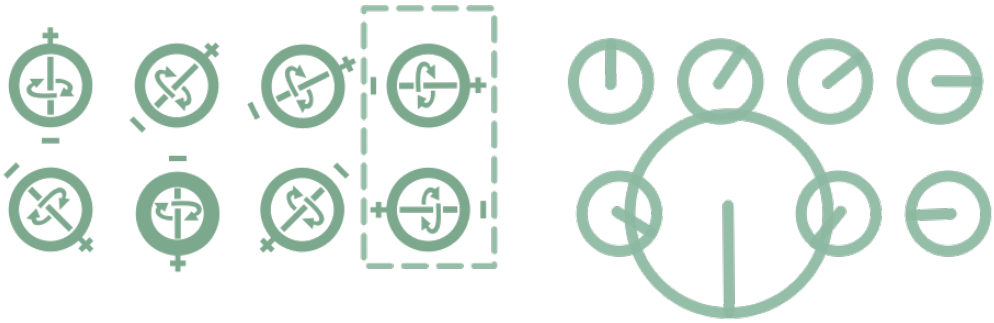
6- Using the mx(r,c) and my(r,c) components, display the new momenta and repeat the cycle for a new time tick s= s+1.

As expected a nodes's energy spreads to its neighbors in a symmetrical fashion. In the simulation four nodes were 'energized' artificially and in isolation as a test pulse – i.e it is not representative of a photon or de-Broglie wave released by or associated with matter. Therefore besides the forward momentum the energy also spreads in the opposite directions – a 'kickback' effect reminiscent of Newton's Third Law 'for every action there is an equal and opposite reaction.

The reason for this in-built property of the simulation will become clearer when a more realistic scenario is implemented with dipolar attraction and repulsion causing symmetrical action all around a node. Results of simulating the transfer of momentum of four test pulses directed at a common center are shown in where the eight states shown. An animated version of this simulation can be seen online [5]. An orderly and symmetrical transfer of momentum, despite the peculiar rules of Eq. (1) and (2) can be considered a preliminary proof of concept of energy transfer in (BU)

5 Animated version of Fig. 1 can be seen online here:
http://vladimirtamari.com/images/bu-basic-animation.gif
https://www.youtube.com/watch?v=NbwBwanXYq4
(Web Acess: July 16, 2016)

III - Limitations of the simulation



a. **b.**

Fig. 2 Comparing Beautiful Universe's(BU) rotating dipolar dielectric nodes – miniature Bloch Spheres - with those used in the simulation. **(a)** *(BU) nodes have + - polarity and rotate in units of Planck's constant (h) . When rotating nodes with opposite polarity lie next to each other (dashed square) they 'stick' together and form matter.* **(b)** *The same nodes using the conventions of this simulation as simple dimensionless vectors. The radius of each circle shows the momentum, but unlike the situation in (BU) in this simulation opposite vectors annihilate each other and cannot show interference e*

The simulation is limited by being 2D, but this does not affect the way unimpeded energy is transmitted across the array. More seriously, as seen in Fig. 2, comparing (BU) nodes to the ones used in the simulation, the simulation nodes do not have the repulsion/attraction properties of dipolar nodes in (BU). There are no interference effects, as when two sine waves add up or subtract when they overlap out of phase, but otherwise continue unaffected. When the four waves of Fig. 1 meet at the center they annihilate each other rather than superimpose as theory demands.

The simulation only considers exchanges between contiguous nodes, while in theory every node is affected by every other node in the array. This first approximation allows the simulation to be made inexpensively in a relatively short program. Another obvious limitation is the lack of quantitative definitions of what is being simulated. In (BU) the nodes rotate with angular momentum in units of Planck's constant (h), and in the vacuum a wave travels at a velocity c. The distances of the nodes and the time-tick value of s in seconds should be defined in a quantitative model. The simulation was implemented using a BASIC app an iPhone and iPad severely limiting the possibilities. A proper 3D simulation with rotating dipolar nodes in the detailed manner of an earlier simulation of three interacting dipoles simulating the Strong Force [6] should be able to show a more realistic wave transfer in the (BU) model.

---

6 Tamari, Vladimir F. , *Three Magnetic Dipoles Provide a Physically Realistic Simulation of the Repulsive-Attractive Nature of the Strong Force and of the Cabibbo Angle (2011)* http://vixra.org/abs/1107.0033 (Web Access: July 6, 2016)

IV- The simulation concept and program.

The BASIC program below is based on the general ideas detailed in Section II, while Fig. 3 shows the exact vector labels and relations used in the program.

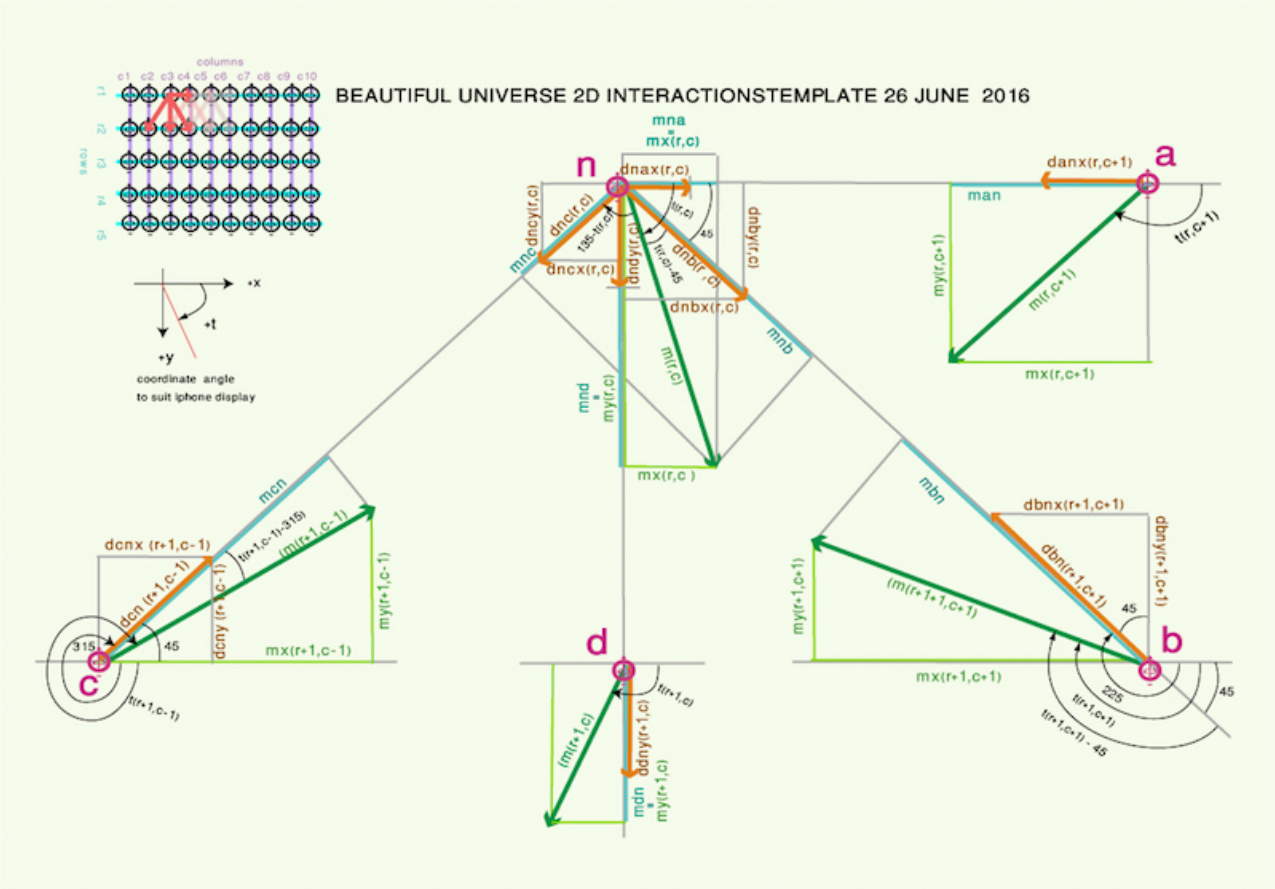


Fig. 3: Diagram of the node geometry and momentum accounting scheme used in the simulation of a 2D square array of nodes (top left). A given node (n) at row (r) and column (c) has initial momentum m(r,c) (green vector) at a phase t(r,c) and contributes some of it to the neighboring nodes a, b, c, d while at the same time it receives momentum from them, according to a specific rule peculiar to the Beautiful Universe model. Only components mna, mnb, mnc, mnd (blue vectors) along the lines joining the nodes na, nb, nc, nd are considered. However n donates only part of those components (orange vectors): for example node (n) contributes to (b) only dnb = mnb / v(1+mnb) where v is a velocity factor taken to be one. At the same time (n) receives momentum dbn from (b), and similarly for the other pairs of nodes adjacent to and directly below n. Finally for each (n) the (x) and y components of all the contributions from and to its neighbors are calculated and stored in memory. The next node cluster in the row then the column is similarly treated systematically. Exchanges along only four lines (red, top left) for each node is sufficient to cover all contiguous unique pairs in the array. After exchanges from all unique pairs of the array are summed up and assigned to each node, the new initial momenta and phases calculated for each node in the next scan of the array.

BASIC program used in the simulation

The following code was edited and displayed using the BASIC! app available from the iOS App Store using an iPad Mini and an iPhone. Figs 1 shows ten screenshots of the iPad tablet as the program was implemented

---

```
REM BEAUTIFUL UNIVERSE JULY 10 2016
REM  SYMMETRICAL TRANSFER 4NODES
REM 2D SQUARE GRID LATTICE nabcd NODES
COLOR 139, 30, 0
BCOLOR  255,255,204
TCOLOR 150,150,150
REM t is node phase ; m is its "density"
sw=ScreenWidth
sh=ScreenHeight '
REM node spacing
n= 20
REM v is a velocity factor
v=1
REM renormalization quotient reevaluated each cycle
ren = 1
REM example n= 24 gives 14 rows 12 columns
r#= INT ((sh- n)/n)
REM #row
c#= INT ((sw - n)/n)
REM # nodes in row
DIM m(r#,c#) ,  t(r#,c#), mx(r#,c#), my(r#,c#)
DIM mna(r#,c#), mnb(r#,c#) , mnc(r#,c#), mnd(r#,c#), man(r#,c#), mbn(r#,c#), mcn(r#,c#), mdn(r#,c#)
DIM dna(r#,c#), dnb(r#,c#) , dnc(r#,c#),  dnd(r#,c#), dan(r#,c#), dbn(r#,c#), dcn(r#,c#), ddn(r#,c#)
DIM   dnby(r#,c#),  dncy(r#,c#), dndy(r#,c#),  dbny(r#,c#),  dcny(r#,c#), ddny(r#,c#)
DIM dnax(r#,c#),  dnbx(r#,c#),  dncx(r#,c#), dbnx(r#,c#),  dcnx(r#,c#), danx(r#,c#)
REM bnay(r#,c#)= dany(r#,c#)= 0. Initial conditions s=1 define angular momentum vector m length with phase angle t at given
r=row, c=column. Set  and display initial vectors m and t of node field. Dots indicate zero momentum nodes. Circles show
momentum of node & its radius m at the phase angle t
REM initial conditions phase t measured from 3 0'clock counterclockwise in RADIANS
j =1/SQR(2)
REM j = SIN or COS (PI/4)


REM nodes at initial conditions.
m(10,19 ) = 500
t(10,19) = PI/2
m(17,27) = 500
t(17,27)= PI
m(17,11) = 500
t(17,11) =  0
m(25,19) = 500
t(25,19) = 3* PI / 2
REM mom is total initial total momentum in system of nodes

FOR r = 1 TO r#-1
    FOR c = 2 TO c#-1
        mx(r,c)= m(r,c)* COS (t(r,c))
        my(r,c)= m(r,c)* SIN (t(r,c))
        mom= mom+ m(r,c)
    NEXT c
NEXT r

REM  display initial and later states
display:
CLS
PRINT s
FOR r = 1 TO r#-1
    y= r*n
```

```
    FOR c = 2 TO c#-1
        x=c*n
        CIRCLE x, y, m(r,c),1
        CIRCLE x,y, 1, 1
        LINE x, y, (x +mx(r,c)), (y + my(r,c)),1
        REM origin is at top left of screen so make +y points down
    NEXT c
NEXT r

REM For each point N Calculating momentum exchange between a set of four unique neighboring node pairs in the square grid.
Each point N has a node A to its right in the same row. In the next row D diagonally to the left , C directly under it in the same
column, and B diagonally to its right. Last column c# and last row # skipped. Scanning the grid systematically  provides the
interactions of each N with all of its immediate neighbors in the grid.
FOR r=1TO r#-1
    FOR c=2 TO c# -1
        REM Momentum exchanges between N & A   (No y components)
        IF m(r,c) = 0 AND m(r,c+1) = 0 THEN
            GOTO skipna
        ENDIF
        mna(r,c) = m(r,c) * COS( t(r,c))
        man(r,c+1) = m(r,c+1)* COS( t(r,c+1))
        dnax(r,c) = mna(r,c) / (v*(1+ABS (man(r,c+1))))
        danx(r,c+1) = man(r,c+1) /(v* (1+ABS (mna(r,c))))
        skipna:
        REM Momentum exchange between Nodes N  & B
        IF m(r,c) = 0 AND m(r+1,c+1) = 0 THEN
            GOTO skipnb
        ENDIF
        mnb(r,c) = m(r,c) * COS (t(r,c)-PI/4)
        mbn(r+1,c+1) = m(r+1,c+1) * COS (t(r+1,c+1)-RAD(45))
        dnb(r,c) = mnb(r,c) /(v*(1+ ABS (mbn(r+1,c+1))))
        dnbx(r,c) = dnb(r,c) * j
        dnby(r,c) = dnb(r,c) * j
        dbn(r+1,c+1) = mbn(r+1,c+1) / (v* (1+ ABS (mnb(r,c))))
        dbnx(r+1,c+1) = dbn(r+1,c+1) * j
        dbny(r+1,c+1) =  dbn(r+1,c+1) * j
        skipnb:
        REM Exchange between Node N  & C
        IF m(r,c) = 0 AND m(r+1,c-1) = 0 THEN
            GOTO skipnc
        ENDIF
        mnc(r,c) = m(r,c) * COS(RAD(135) - t(r,c))
        mcn(r+1,c-1) = m(r+1,c-1) * COS((t(r+1,c-1) - RAD(315)))
        dnc(r,c) = mnc(r,c) / (v*(1+ ABS (mcn(r+1,c- 1))))
        dncx(r,c) =  - dnc(r,c) * j
        dncy(r,c) = dnc(r,c) * j
        dcn(r+1,c-1) = mcn(r+1,c-1) / (v*(1+ ABS (mnc(r,c))))
        dcnx(r+1,c-1) =dcn(r+1,c-1) * j
        dcny(r+1,c-1) = - dcn(r+1,c-1) * j
        skipnc:
        REM Momentum exchanges between N & D  (No x components)
        IF m(r,c) = 0 AND m(r+1,c) = 0 THEN
```

```
            GOTO skipnd
        ENDIF
        mnd(r,c) = m(r,c) * SIN( t(r,c))
        mdn(r+1,c) = m(r+1,c)* SIN( t(r+1,c))
        dndy(r,c) = mnd(r,c) / (v*(1+ABS (mdn(r+1,c))))
        ddny(r+1,c) = mdn(r+1,c) /(v* (1+ABS (mnd(r,c))))
        skipnd:
    NEXT c
NEXT r

REM CALCULATING NEW MOMENTA AFTER EXCHANGES OF ENTIRE NODES
FOR r=1 TO c#-1
    FOR c=2 TO c# -1
        REM Cumulative X Y of Momentum for node N
        mx(r,c)= mx(r,c) +danx(r,c+1) +dbnx(r+1,c+1) + dcnx(r+1,c-1) -dnax(r,c) - dnbx(r,c) - dncx(r,c)
        my(r,c) = my(r,c) + dbny(r+1,c+1) + dcny(r+1,c-1) +ddny(r+1,c) - dnby(r,c) - dncy(r,c) -dndy(r,c)
        REM Cumulative X Y of Momentum for node A
        mx(r,c+1)= mx(r,c+1)+ dnax(r,c) - danx(r,c+1)
        REM Cumulative X Y of Momentum for node B
        mx(r+1,c+1)= mx(r+1,c+1) +dnbx(r,c) -  dbnx(r+1,c+1)
        my(r+1,c+1) = my(r+1,c+1)  +dnby(r,c)  - dbny(r+1,c+1)
        REM Cumulative X Y of Momentum for node. C
        mx(r+1,c-1)= mx(r+1,c-1) +dncx(r,c)  - dcnx(r+1,c-1)
        my(r+1,c-1) = my(r+1,c-1) +dncy(r,c) - dcny(r+1,c-1)
        REM Cumulative X Y of Momentum for node D
        my(r+1,c)= my(r+1,c) +dndy(r,c) - ddny(r+1,c)
    NEXT c
NEXT  r

REM UPDATED MOMENTUM AND PHASE FOR NEXT ROUND
FOR r = 1 TO r#-1
    FOR c = 2 TO c#-1
        IF mx(r,c)=0 AND my(r,c)=0 THEN
            GOTO skipangle
        ENDIF
        IF mx(r,c)=0  AND my(r,c) < 0 THEN
            t(r,c)=  3*PI/2
            GOTO skipangle
        ENDIF
        IF mx(r,c)=0 AND my(r,c) > 0 THEN
            t(r,c)=PI/2
            GOTO skipangle
        ENDIF
        IF mx(r,c)<0 AND my(r,c) =0 THEN
            t(r,c)=  PI
            GOTO skipangle
        ENDIF
        IF mx(r,c)>0  AND my(r,c) = 0 THEN
            t(r,c)=0
            GOTO skipangle
        ENDIF
        t(r,c) =  ATN( my(r,c)/mx(r,c))
```

```
        IF mx(r,c)<0 AND my(r,c) <0 THEN
            t(r,c)= t(r,c) +PI
        ENDIF
        IF mx(r,c)<0 AND my(r,c) >0 THEN
            t(r,c) = -PI + t(r,c)
        ENDIF
        skipangle:
        m(r,c) =((mx(r,c)^2+ my(r,c)^2)^0.5 )
        momtot = momtot + m(r,c)
    NEXT c
NEXT r

REM renormalize all momenta to conserve total momentum
ren= mom/momtot
FOR r = 1 TO r#-1
    FOR c = 2 TO c#-1
        mx(r,c)=mx(r,c)*ren
        my(r,c)=my(r,c)* ren
        m(r,c)= m(r,c)*ren
    NEXT c
NEXT r

REM count cycle number
s=s+1
momtot=0
GOTO display
```