RESEARCH LOG - WEEK 2

THOMAS HOWSON

1. Coronal Loop Tracing

If during the first week the author was more concerned with the logistics of life in Bozeman, MT, it is safe to say that research has begun in earnest over the past few days. Monday witnessed the culmination of the loop tracing that was started during the previous week. It seemed to be the case that once a loop had been ended, it could be brought back from the dead with a click of life. In less obscure words, the author observed that a loop that had been terminated many snapshots earlier, would sometimes reappear if subsequent loops were terminated in the vicinity of the original loop. This was not immediately obvious as the resurrected loop would only have an extended life equivalent to the time difference between its original point of death and the moment the new loop was ended. For example, a loop ended on frame 660 could be awarded a stay of execution until say, frame 730 if at this later time, a different loop was ended close to the original loop. Whilst this phenomena must be a fault of the code, the author did not try to resolve this issue since the code appeared far too complicated to be easily rectified. Instead, after much flicking forwards and backwards through the slides, the finished article did seem to mimic the evolution of coronal loops fairly accurately.

2. Coding - The Beginnings

Upon completion of the loop tracing exercise a '.sav' file was produced that contained an array of structures describing the properties of each loop that had been plotted. It wasn't immediately obvious, at least not to the author's untrained eye, as to what was represented by each list of numbers. Consequently, a reasonable amount of time was expended upon the task of interpreting the array of structures contained within the .sav file. Fortunately, this proved to be time well spent as it made subsequent manipulation of this data both easier and quicker. Following this exercise, any degenerate/erronous loops were deleted using a code that identified any loops that had been traced in the wrong location and any loops that had been deleted immediately after creation.

3. Loop Analysis

Once the unwanted loops had been discarded, an initial analysis of the remaining loops was undertaken. Plots were generated showing the loop length as a function of its initial appearance time (Fig. 1) and the loop lifetime as a function of its initial appearance time (Fig. 2). For the second of these plots any loops that had not been terminated after the one thousandth snapshot were discounted since these were associated with an essentially infinite lifetime.

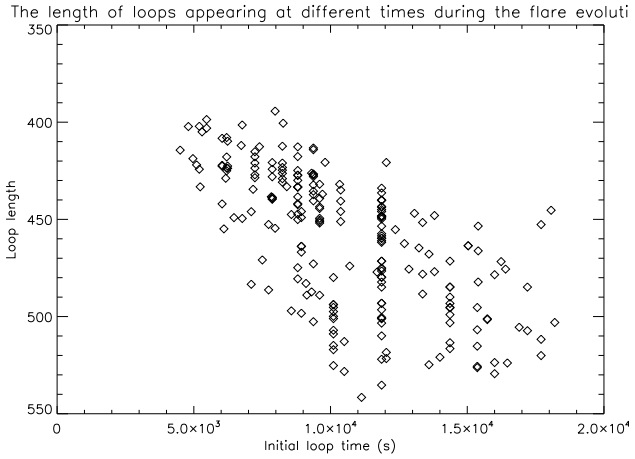

Figure 1. Loop length against initial time of loop appearance

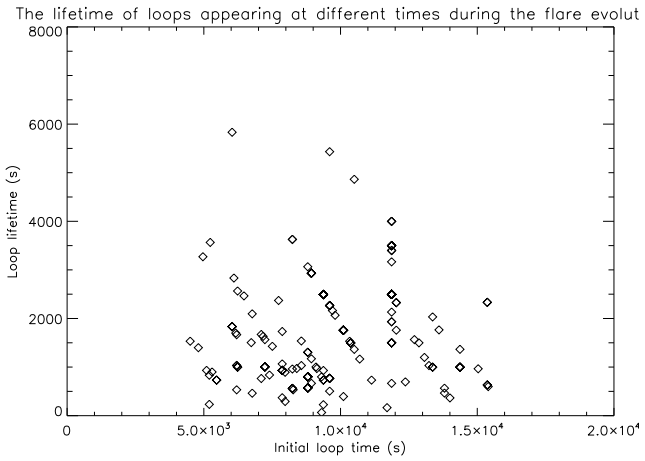

Figure 2. Loop lifetime against initial time of loop appearance

4. Lectures and Seminars

This week the REU students attended lectures on heat transfer processes inside the sun and on the sun's magnetic field, delivered by Dr Dana Longcope and Dr Sarah Jaeggli respectively. This author was certainly battling to recall the Physics formulae that have so long been absent from his education. The basic reasons behind the existence of the sun's dynamo were especially interesting and an understanding of these is certainly fundamental to the modelling solar theoriests undertake. Despite the author's previous experience of simple modelling of solar structures, an overview of the physical processes that drive the formation of these phenomena was certainly valuable. Wednesday saw a seminar presented by Dr Ying Li (of Nanjing University) on the topic of spectroscopy techniques used in solar flare observation. The use of the Doppler Effect to determine the motion of plasma during solar flare events was discussed. Furthermore, the Zeeman effect (the splitting of spectral lines in the presence of magnetic fields) was considered. This author was previously unfamiliar with this effect and so found this section of the seminar fascinating.

5. Appendix

5.1. **Original Code.** Whilst the code presented below is undoubtedly neither impressive nor effecient, it represents the first real coding the author has ever completed and so it is included within this log.

```
pro newloops ;program for finding degenerate loops and plotting Fig. 1 and Fig. 2
restore, 'loops171v3.sav' ;the file containing information on the loops traced previously
newloops = dindgen(264) ;create array to contain the lifetime of each loop
for i = 0, 263 do begin
newloops[i] = loops[i].(1)-loops[i].(0) ;adds lifetime to newloops
endfor
j = 264
for k = 0, 263 do begin
if newloops[k] le 0 then begin ;defines a degenerate loop (by time)
j = j - 1 ; finds number of degenerate loops
endif
endfor
m = 0
nloops = indgen(j) ;defines new array of suitable size
for n = 0, 263 do begin
if newloops[n] gt 0 then begin
nloops[m] = n ;adds index of non-degenerate loops to the array nloops
m = m + 1
endif
endfor
p=n_elements(nloops)
xin=fltarr(p) ;defines an array for the initial x-coordinate of each loop
yin=fltarr(p) ;defines an array for the initial y-coordinate of each loop
```

```
for n=0, p-1 do begin
xin[n] = loops[nloops[n]].(3)[0] ;adds the inital x-coordinate of each loop to the xin array
yin[n] = loops[nloops[n]].(3)[1] ;adds the initial y-coordinate of each loop to the yin array
endfor
j1=p
for i = 0, p-1 do begin
if (yin[i] + (3/5)*xin[i]) lt 310 then begin ;defines degenerate loops by space (obtained using plot)
j1 = j1 - 1 ;finds number of non-degenerate loops (by space)
endif
endfor
nloops2 = indgen(j1) ;defines array for non-degenerate loops
count2 = 0
for i = 0, n_elements(nloops)-1 do begin
if (yin[i]+(3/5)*xin[i]) ge 310 then begin
nloops2[count2] = nloops[i] ;adds index of non-degenerate loops to the array nloops2
count2 = count2 +1
endif
endfor
nl= n_elements(nloops2)
yin2=fltarr(nl) ;creates an array for the initial y coordinate of each loop
xin2=fltarr(nl) ;creates an array for the initial x coordinate of each loop
for i = 0, nl-1 do begin
j = nloops2[i]
xin2[i] = loops[i].(3)[0] ;adds the initial x coordinate of each loop to the array xin2
yin2[i] = loops[i].(3)[1] ;adds the inital y coordinate of each loop to the array yin2
endfor
finalloops = loops[nloops2] ;generates a structure containing all non-degenerate loops
save, finalloops, filename = "finalloops.sav" ;saves this structure in the current directory
looplen = dblarr(nl) ;creates an array for the length of each loop
initime = dblarr(nl) ;creates an array for the time of initial appearance of each loop
for i = 0, nl-1 do begin
suml = 0
k=finalloops[i].xy
for j = 0, 95 do begin
suml = suml + sqrt( (k[1,j+1]-k[1,j])^2 + (k[0,j+1] - k[0,j])^2); findsthelengthofeachloop
endfor
looplen[i] = suml ;adds the length of each loop to the array looplen
initime[i] = finalloops[i].ti*10000 - (1.15*(10.0^5.0)); addstheappearancetime(insecondsafteranarbitrary0time)
endfor
a = min(looplen) ;a,b,c,d and e used in generating plots
b = max(looplen)
c = min(initime)
d = max(initime)
looplife = (finalloops.tf-finalloops.ti)*10000 ;generates an array containing the lifetimes of each loop
e = min(looplife)
f = 6000
set_plot, 'PS' ;generates a suitable plotting environment and then the required plots are generated.
Device, /color, Filename = 'Loop_length.eps'
```

plot, initime, looplen, psym =4, xtitle = "Initial loop time (s)", ytitle = "Loop length", xrange = [c-0.1,d+0.1], yrange = [b-0.1, a+0.1], title = "The length of loops appearing at different times during the flare evolution"

Device, /close

set_plot, 'PS'

Device, /color, Filename = 'Loop_lifetime.eps'

plot, initime, looplife, psym = 4, xtitle = "Initial loop time (s)", ytitle = "Loop lifetime (s)", title = "The lifetime of loops appearing at different times during the flare evolution", xrange = [c-0.1, d+0.1], yrange = [e -0.1, f +0.1]

Device, /close

end