Subject: Some questions of efficiency when matching items in lists
Posted by Matt Francis on Fri, 01 Feb 2013 00:05:40 GMT
View Forum Message <> Reply to Message

I have a couple of questions about how to efficiently match items in lists. There are two operations that are done many thousands of times in my processing and are causing a bottleneck. Even small improvements would be welcome.

The first issue is writing information with an associated time to a daily file containing all the data for the day. The file is a simple CSV format with the time of day then the value as the two columns. The processing occurs in near real time, but the complication is that 'older' values often need to be updated as well as the latest value needing to be written, so I can't do a simple append. The current approach for adding a single value at some time is this:

```
timeS = <string representation of timestamp of this data value>
; Read existing data
nlines = file_lines(fname)
fdata = strarr(2,nlines)
sdata = strarr(nlines)
openr,lun,fname,/get_lun
readf,lun,sdata
free_lun,lun
for j=0,nlines-1 do fdata[*,j] = strsplit(sdata[j],' ',/extract)
; Check for overwrite or append
indx = where(timeS eq fdata[0,*],count)
if count eq 1 then fdata[1,indx] = new_data else fdata=[[fdata],[timeS,new_data]]
; Write out to temp file
openw,lun,tfname,/get_lun
for j=0,n_elements(fdata)/2-1 do printf,lun,fdata[0,j],' ',fdata[1,j]
free_lun,lun
; Use file system to overwrite file
file_move,tfname,fname,/over
```

The other related issue is that I later need to collate these files for many different data sources into a single handy structure for analysis. I read each file into an array of {time:0.,value:} structures and combine into a single {time:0., a:0., b:0., ...} structure array. The different sources may have missing data points, so the times need to be matched up carefully. At the moment I do something like

```
data_add = <structure array for a single data source>
data = <structure array for the collated data from all sources>
for i=0,n_elements(data_add)-1 do begin
   indx=where(data_add[i].time eq data.time,count)
   if count eq 1 then data[indx].<this source tag> = data_add[i].value
```

Can the outer loop be removed by some single operation that matches all the elements in data_add.time to the elements in data.time ?