Subject: Re: Multi-column sort
Posted by Percy Pugwash on Mon, 05 Mar 2012 10:53:17 GMT
View Forum Message <> Reply to Message

Thanks very much. I had looked at Craig Markwardt's multisort, but it didn't quite do what I wanted (the number of columns to sort by was limited and could not easily be changed at run-time).

I've not looked at bsort yet, but will have a look, thanks. My only concern is that I'm aware that bubble sort is usually much slower than quicksort, which I believe sort() uses... Might still be worth it though.

The solution I came up with is below, in case anyone is interested. Please bare in mind that it has not been properly tested, but seems to be working.

P

```
function sort_strcolumns, strtable, indices
   maxlen = max(strlen(strtable[abs(indices),*]))
   ncols = n_elements(indices)
   nrows = (size(strtable,/dim))[1]
   sortlist =  reform(string(strtable[abs(indices),*],f='(a-'+string(maxlen
,f='(i0)')+')'),ncols,nrows,/overwrite)
   sortlist = reform(byte(sortlist),maxlen*ncols,nrows,/overwrite)
   for i=0,n_elements(indices)*maxlen-1 do sortlist[i,*] *= (-1)^(indices[i/maxlen] lt 0)
   return, sort(string(sortlist))
end
```

On Friday, 2 March 2012 12:26:00 UTC, Gianguido Cianci wrote:
> Firstly, sort() does not maintain the order of identical elements so I'd use bsort() which you can find online somewhere, can't remember where... I believe it has a /reverse or /invert option, not at my computer right now.
>
> Secondly, you should bsort columns in increasing order of importance, with the most important sort last.
>
> I have a dumb for-loop procedure that does that. This requires multiple searches through the array, which might not be optimal, but once you write it, you'll never use sort instead of bsort again :-)
>
> G