
Subject: Re: QUESTION: special characters in postscript output

Posted by [sterne](#) on Sun, 14 Mar 1993 06:12:04 GMT

[View Forum Message](#) <> [Reply to Message](#)

In article <1993Mar13.015020.29487@bnlux1.bnl.gov> rivers@bnlux1.bnl.gov (mark rivers) writes:

```
> ; This program section defines some strings which contain both English
> ; and Greek characters. It treats PostScript and all other devices
> ; differently. The strings are "K (alpha)", "K (beta)", etc. These are
> ; symbols for x-ray emission lines. This works in IDL 3.0. I am not sure
> ; about PV-WAVE.
>
```

This does not work in PV-WAVE, at least at version 3.10. The !9, !X and !D are not converted in any way and are printed out as literal characters. I don't know if the more recent releases of WAVE have implemented this badly needed feature. If they have, I'd be interested in hearing about it, as it might change my mind about buying into their maintenance program.

In the meantime, it is possible to get some of the same capability in PV-WAVE by modifying the postscript prolog file. A while back, Robert Boughner sent me a modified postscript prolog file he had made for IDL which converted characters within < > to greek. I made the trivial modifications for PV-WAVE. Here is the patch. Apply it to the wave_prolog.ps file which is in the WAVE_DIR directory, usually something like /usr/local/pvi/wave on a U**x box. Somebody who speaks postscript well enough might like to make it more compatible with the current IDL behavior. Please post if you do!

```
*** wave_prolog.ps.orig Sat Mar 13 22:58:39 1993
```

```
--- wave_prolog.ps Sat Mar 13 23:02:04 1993
```

```
*****
```

```
*** 30,36 ***
```

```
    % x y string size orientation alignment T
```

```
    %
```

```
    % Define our dictionary:
```

```
! /$T_DICT 20 dict def
```

```
    %
```

```
    % Define the procedure
```

```
    %
```

```
--- 30,36 ----
```

```
    % x y string size orientation alignment T
```

```
    %
```

```
    % Define our dictionary:
```

```
! /$T_DICT 26 dict def
```

```
    %
```

```

% Define the procedure
%
*****
*** 38,43 ***
--- 38,46 ---
% The character with an octal code of 330 starts superscripting,
% 331 starts subscripting, and 332 reverts to normal.
%
+ % Modified by REB on December 3, 1991 to handle switching to Greek
+ % characters which are indicated by enclosing them in < > characters
+ %
$T_DICT begin      % Use this dictionary
/align exch def    % Get parameters
/orien exch def
*****
*** 53,84 ***
/SUBS 8#331 def    % Escape code for subscript
/NORM 8#332 def    % Escape code for normal text
/SCRIPTWID 0.7 def % Size of super & subscripts
! /OFFSET 0.6 def   % Offset from baseline of sub & super scripts
%
orien rotate      % Rotate for text orientation
size dup scale    % Set global scale factor
%
/orien false def   % Will be true if there are subs/super scripts

! thestring {      % Loop for each character
!   /charcode exch def % Save code
!   charcode SUPER eq charcode SUBS eq or
!   { /chsize SCRIPTWID def % if super | subs, make smaller
!     /orien true def }      % Set flag indicating scripts
!   { charcode NORM eq      % Back to normal
!     { /chsize 1.0 def
!       /orien true def }
!     { ( ) dup 0 charcode put % Make a 1 character string for the char
!       stringwidth pop chsize mul xsize add % Get width & accum
!       /xsize exch def      % Save back in xsize
!     } ifelse }
!   ifelse
! } forall
xsize align mul neg 0 rmoveto % Do the adjustment for alignment
%
orien      % Need to do font stuff?
{
- /regularfont currentfont def % save current font
/fractionfont currentfont [ SCRIPTWID 0 0 SCRIPTWID 0 0 ]
  makefont def % Make a new font for subscripts & superscripts
gsave          % Get height of numeral 1 in current font

```

```

--- 56,116 ---
/SUBS 8#331 def      % Escape code for subscript
/NORM 8#332 def      % Escape code for normal text
/SCRIPTWID 0.7 def   % Size of super & subscripts
! /SYMST (<) 0 get def % Start of symbol fonts
! /SYMEND (>) 0 get def % End of symbol fonts
! /SKPNXT (/) 0 get def % Escape character
! % "FontSize" determines size of currentfont and leaves it on stack
! /FontSize
! { currentfont /FontMatrix get 0 get 0.001 div } def
%
orien rotate        % Rotate for text orientation
size dup scale      % Set global scale factor
%
/orien false def    % Will be true if there are subs/super scripts
+ /regularfont currentfont def % save current font
+ /symfont /Symbol findfont FontSize scalefont def %symbol fonts
+ /NOSKIP true def
+ /size {
+   ( ) dup 0 charcode put % Make a 1 character string for the char
+   stringwidth pop chsize mul xsize add % Get width & accum
+   /xsize exch def % Save back in xsize
+   } def
+ /TestForSkp { % char on stack, on return, leave a Boolean
+   /chr exch def
+   chr SKPNXT eq chr SYMST eq or chr SYMEND eq or not } def
+ /StrLen { thestring length 1 sub } def

!
! 0 1 StrLen { % Loop for each character
!   /indx exch def
!   /charcode thestring indx get def %Get charcode
!   charcode SKPNXT eq NOSKIP and % Chk if escape character
!   % if yes, don't treat the next character in a special manner
!   % if last character is "/", then just return " "
!   {/NOSKIP thestring indx 1 add dup StrLen gt
!     {pop pop ( ) 0}if get TestForSkp def
!     NOSKIP {size}if }
!   {charcode SYMST eq NOSKIP and % go into symbol fonts?
!     {symfont setfont}
!     {charcode SYMEND eq NOSKIP and % end symbol fonts?
!       {regularfont setfont}
!       {charcode SUPER eq charcode SUBS eq or
!         { /chsize SCRIPTWID def % if super | subs, make smaller
!           /orien true def } % Set flag indicating scripts
!         {charcode NORM eq % Back to normal
!           { /chsize 1.0 def
!             /orien true def }

```

```

!      { size /NOSKIP true def
!      } ifelse
!      } ifelse
!      } ifelse
!      } ifelse
!      } ifelse
!      } for
      xsize align mul neg 0 rmoveto % Do the adjustment for alignment
%
      orien      % Need to do font stuff?
      {
        /fractionfont currentfont [ SCRIPTWID 0 0 SCRIPTWID 0 0 ]
        makefont def % Make a new font for subscripts & superscripts
        gsave      % Get height of numeral 1 in current font
*****
*** 87,109 ****
      grestore
    } if
%
    /xsize 0 def      % Starting out a 0 baseline
!   thestring {      % Now draw each character
!     /charcode exch def % char code
      charcode SUPER eq
!     { 0 OFFSET height mul dup /xsize exch def % Up for superscripts
!       rmoveto fractionfont setfont } % Set this font
!     { charcode SUBS eq % Down for subscripts
!       { 0 OFFSET height mul neg dup /xsize exch def rmoveto
!         fractionfont setfont }
!       { charcode NORM eq
!         { 0 xsize neg rmoveto % Back to normal
!           regularfont setfont } % Baseline & size
!         { ( ) dup 0 charcode put show } % Otherwise show it
!       } ifelse }
!     } ifelse }
!   } ifelse }
!   forall
      grestore      % Restore original context
      end          % Restore original dictionary
    } bdef
--- 119,164 ----
      grestore
    } if
%
+ /orien false def
  /xsize 0 def      % Starting out a 0 baseline
! /savfont currentfont def
! /NOSKIP true def
! /PrntIt {

```

```

!   orien {fractionfont}{regularfont}ifelse setfont
!   ( ) dup 0 charcode put show % Otherwise show it
!   } def
!
! 0 1 StrLen {      % Now draw each character
!   /indx exch def      %get index value
!   /charcode thestring indx get def % and the char code
!   charcode SUPER eq
!   { 0 0.6 height mul dup /xsize exch def rmoveto % Up for superscripts
!   /orien true def }
!   {charcode SUBS eq % Down for subscripts
!   { 0 0.3 height mul neg dup /xsize exch def rmoveto
!   /orien true def }
!   {charcode SKPNXT eq NOSKIP and
!   {/NOSKIP thestring indx 1 add dup StrLen gt
!   {pop pop ( ) 0}if get TestForSkp def
!   NOSKIP {PrntIt}if }
!   {charcode SYMST eq NOSKIP and
!   {symfont setfont /regularfont currentfont def
!   orien {/fractionfont currentfont [ SCRIPTWID 0 0 SCRIPTWID 0 0 ]
!   makefont def }if }
!   {charcode SYMEND eq NOSKIP and
!   {/regularfont savfont def
!   orien{/fractionfont savfont [ SCRIPTWID 0 0 SCRIPTWID 0 0 ]
!   makefont def } if }
!   {charcode NORM eq
!   { 0 xsize neg rmoveto /orien false def } % Back to normal
!   { PrntIt /NOSKIP true def
!   } ifelse
!   } ifelse
!   } ifelse
!   } ifelse
!   } ifelse
!   } ifelse
!   } for
!   grestore      % Restore original context
!   end           % Restore original dictionary
! } bdef

```

Enjoy,
Phil

--

Philip Sterne | sterne@dublin.llnl.gov
Lawrence Livermore National Laboratory | Phone (510) 422-2510
Livermore, CA 94550 | Fax (510) 422-7300
