PharmaSUG 2025 Paper AP-006

Calculating the Physical Length of a PDF File String with 'Times New Roman' Font by SAS

William Wu, Taiho Oncology, Inc., Pleasanton, CA

Steven Li, Medtronic PLC, Mounds View, MN

ABSTRACT

When using the PROC REPORT procedure with ODS PDF statement for output table/listing generation, people often wish to know the 'Physical Length' of a string, instead of the number of characters in the string. The physical length of a string could be used to decide the column's minimum width without word wrapping up, or without overlapping to the next column. This paper created a custom SAS function 'pwidth()' through PROC FCMP to calculate the physical length of a string on font 'Times New Roman' in PDF file. It may be useful for generating a nice FDA's BIMO (Bioresearch Monitoring) PDF Listing.

INTRODUCTION

When exporting SAS output to a PDF file with the SAS Output Delivery System (ODS) and a powerful procedure PROC REPORT, it's preferable to define a standard font such as 'Times New Roman' and set a suitable column width for each column. The column width setting depends on the physical length of strings/words inside of the column's cells. Therefore, it becomes necessary to obtain the physical width of a string. On the column width setting, oversize will occupy limited space, and undersize will cause word wrap up.

This paper illustrates how to calculate the physical length for a string in PDF file with font name as 'Times New Roman', the font weight as either 'Normal' or 'Bold', and font style as 'Roman' (instead of 'Italic'). The paper discusses the physical length on ODS PDF statement with option 'dpi=1440' for different font size (mainly from font size 1 pt to 12 pt).

A string is comprised of many characters, and there are 95 different characters (from ASCII code 32 to ASCII code 126). Each character might have its own physical width, or sometimes two or more characters share the same physical width. Additionally, the physical width of all 95 characters should be calculated individually for both font weights 'Normal' and 'Bold'.

All demonstrations and sample code in this paper use version 9.4 (TS1M3) of SAS on a Windows platform to generate a PDF file and displayed it by application 'Adobe Acrobat X Pro'.

The Physical Length calculation in a PDF file is different with the Physical Length calculation in an RTF file. For the Physical Length in an RTF file, please read our previous paper "Calculate Physical Length of String in RTF File with 'Times New Roman' by SAS" [1].

The biggest difference between ODS PDF and ODS RTF statements is that in PDF destination, the Generalized Physical Length is included in 'Physical Length' and 'Vision Physical Length' (short for 'Vision Length').

THE CONCEPTS OF PHYSICAL LENGTH AND VISION LENGTH

To generate a PDF output file by procedure 'PROC REPORT' with ODS PDF statement, a typical code is as following:

```
ods _all_ close;
ods pdf file="<file name>.pdf" style=<style name> dpi=1440 compress=0;

proc report data=<dataset name> nowd;
```

```
    column <variable list>;
    define <first variable> /display style=[just=<l|c|r> cellwidth=<Wc>pt] '<label>';
    … …
run;

ods pdf close;
ods listing;
```

In the code as above, option ‘style=<style name>’ is for assigning a suitable style to the output PDF file, include the font name, font size and font weight to the strings, and even the cell’s format setting such as the value to ‘cellpadding’, ‘borderspacing’, and ‘borderwidth’.

In the ‘ODS PDF’ statement, the option ‘dpi=’ is the setting to specify the resolution for the output file. ‘dpi=1440’ means the resolution is 1440 dot per inch. Or in other words, the resolution could be converted as 0.05 pt per dot (1 inch = 72 pt). Different value setting on ‘dpi=’ will cause different value on the physical length and vision length calculation. So, in this paper, it’s always kept at the setting option of ‘dpi=1440’.

The option ‘compress=0’ is for generating an uncompressed PDF output file (named "<file name>.pdf") for convenient reading and extraction of the detail of column’s information (such as x-y coordinates) through ‘Notepad’ software. For a detailed explanation, please reference our previous paper “Importing Data Directly from PDF into SAS® Data Sets” [2]. The code in paper [2] demonstrated how to extract and convert the data from the uncompressed PDF file into a SAS data set. Those code is useful in the present paper in multiple macros on extracting and converting, macro %location in APPENDIX 2 implemented it.

The option ‘cellwidth=<Wc>pt’ in ‘define’ statement of PROC REPORT is for setting the width of each column.

Before discussing the Physical/Vision Length of a string, a macro %style in APPENDX 2 should be introduced. The user interface is as following:

%style(stylen=, boldn=, frame=, rules=, fontsize=, margin=, cellpadding=, borderspacing=, borderwidth=)

The macro %style could generate a new style with all font name as “Times New Roman”. Then, the style could be used by ODS PDF statement with option ‘style=’.

An PDF output generated by macro %Char_y_m in APPENDIX 2 is as following:

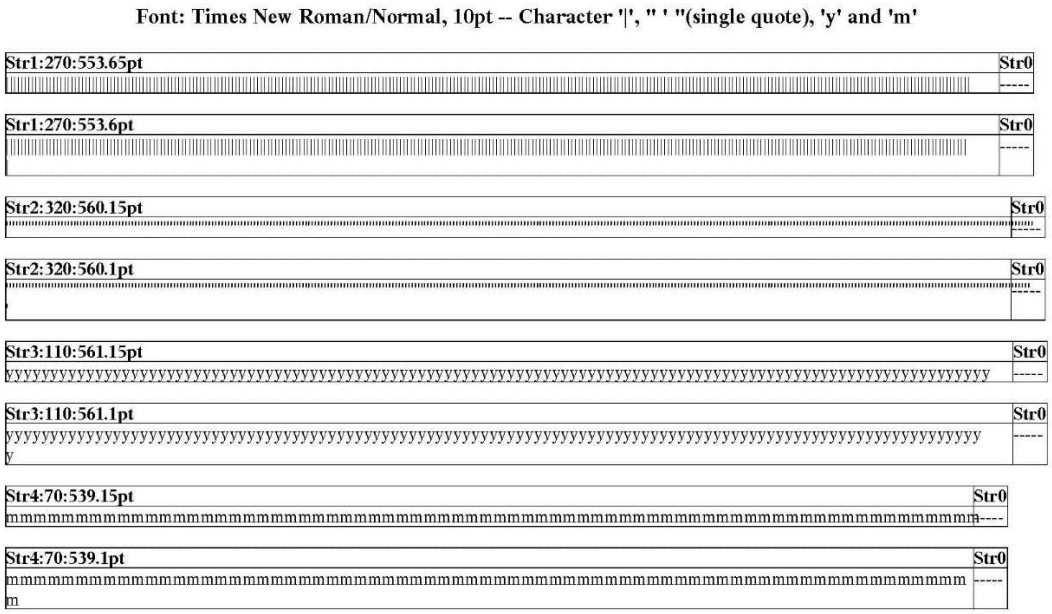

Figure 1: Macro %Char_y_m illustrated the difference between Physical Length and Vision Length.

In Figure 1, the program setting in macro %Char_y_m on the style is:

frame=2 (as frame=box), cellpadding=0pt, borderspacing=0.15pt, borderwidth=0.15pt

The variable ‘Str1’ is with 270 repeated single character ‘|’s and is displayed as column width 553.65 pt (in Row 1) without wrap up. But displayed as column width 553.6pt (in Row 2) with wrap up. So, minus 0.15pt (as the value of ‘borderwidth’), the rest of column width 553.5 pt is referred to as the **critical column width** for 270 continuous characters ‘|’s here. The Vision Width, however, is shorter than that value. There is a space not used at the right end of the column (if used, at least 8 characters ‘|’ should be filled out for the space).

And variable ‘Str2’ is with 320 repeated single character “ ‘ “ (single quotes) and displayed as column width Wc=560.15 pt (in Row 3) without wrap up, but column width 560.1 pt (in Row 4) with wrap up. So, minus 0.15 pt (as the value of ‘borderwidth’), the rest of column width 560 pt is referred to as the **critical column width** for 320 repeated single character “ ‘ “. However, the Vision Width of 320 repeated single character “ ‘ “ is longer than the value 560pt. The string therefore went over to the right end of the column for as many as 7 character “ ‘ “ without wrap up.

For letter ‘y’, the Vision Width is shorter than the Physical Width. And for letter ‘m’, the Vision Width is longer than the Physical Width (see Row 5 to Row 8 in the Figure 1).

Therefore, some characters' Physical Width and Vision Width could differ. Thus, the concepts of ‘Physical Width’ and ‘Vision Width’ of a string have been distinguished. In the sections that follows they will be discussed individually.

COLUMN WIDTH AND FRAME ATTRIBUTES, CODING AND ACTUAL WIDTH

As mentioned above, we called ‘Wc’ as the ‘Coding Column Width’ and used style element ‘cellwidth=<Wc>pt’ in STYLE= option for DEFINE statement on PROC REPORT. As a result, ODS PDF statement set an ‘Actual Column Width‘ as <Wa>pt. Sometimes Wa=Wc, while sometimes both values are not equal.

Determining the value of Wa (‘Actual Column Width’) on macro %WC_WA in “APPENDX 2”, an output PDF file is illustrated figure 2.

WC_WA.pdf

1:499.97
0000000000000000000000000000000000000000000000000000000000000000000000000 |
0

2:499.98
0000000000000000000000000000000000000000000000000000000000000000000000000|

3:499.99
0000000000000000000000000000000000000000000000000000000000000000000000000|

4:500
0000000000000000000000000000000000000000000000000000000000000000000000000|

5:500.01
0000000000000000000000000000000000000000000000000000000000000000000000000|

6:500.02
0000000000000000000000000000000000000000000000000000000000000000000000000|

7:500.03
0000000000000000000000000000000000000000000000000000000000000000000000000|

Figure 2: PDF file ‘WC_WA.pdf’ with 7 lines for ‘coding column width’ 499.97 to 500.03pt

Instead of opening the file by ‘Adobe Acrobat X Pro’, we could open it by ‘Notepad’ to show the following:

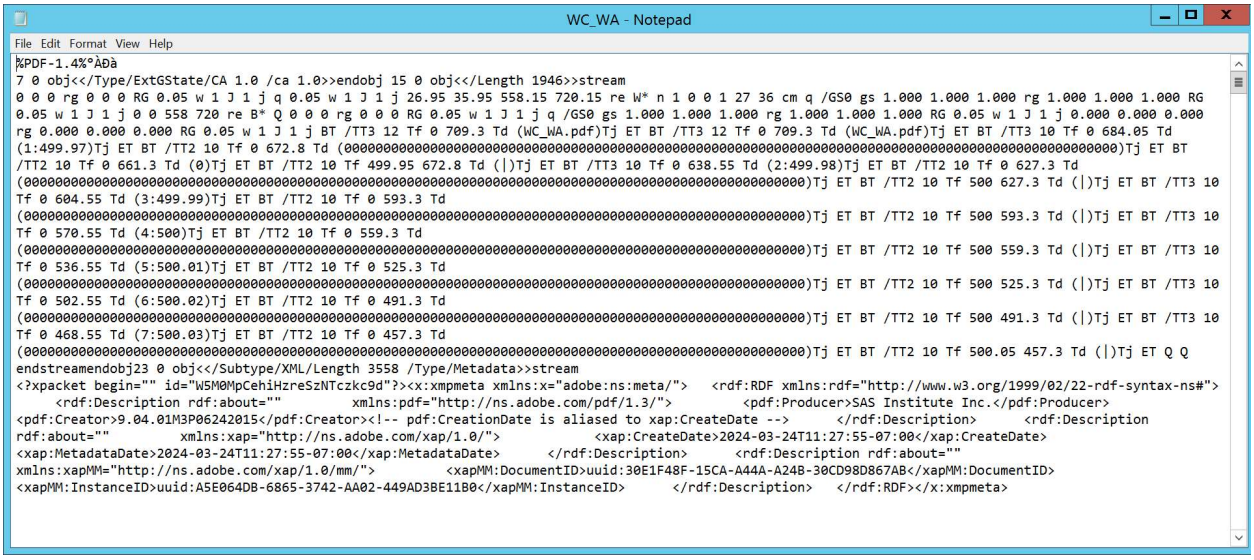

Figure 3: Open PDF file ‘WC_WA.pdf’ using Notepad to read the location information for the texts.

Figure 3 shows ‘(<text>)’ for each text in the ‘cell’, or for the ‘title’. And before that part, it shows ‘Tf <x-coord> <y-coord> Td’ for related (x,y) coordinate. For example, a string as ‘Tf 0 684.05 Td (1:499.97)Tj’ in Notepad software, tell us the text ‘1:499.97’ will be displayed into the PDF file, and the (x,y) coordinate is (0, 684.5). The (x,y) coordinate is counted from bottom left corner of the page (this is as the origin point (0,0)), and the unit is ‘pt’. <x-coord> and <y-coord> keep two digits as the maximum number of digits after the decimal point. According to our previous paper [2] (“Importing Data Directly from PDF into SAS® Data Sets”), we coded a macro %Location in ‘APPENDIX 2’ to extract ‘text’ and related (x,y) coordinate’ information. The macro %WC_WA in ‘APPENDIX 2’ provided the relationship between <Wc> and <Wa> as shown in Figure 4:

	K1	WC	WA	FL
1	1	499.97	499.95	*
2	2	499.98	500	
3	3	499.99	500	
4	4	500	500	
5	5	500.01	500	
6	6	500.02	500	
7	7	500.03	500.05	

Figure 4: Relationship between Wc and Wa. Wa rounded by Wc to nearest multiple of 0.05pt

From Figure 4 now, we can see, in PROC REPORT with ODS PDF, any coding setting to the column width ‘Wc’ has been converted to the actual column width ‘Wa’ by rounding the ‘Wc’ to the nearest multiple of 0.05 (unit ‘pt’) while ODS PDF statement has option ‘dpi=1440’. In a formula:

$$Wa = round(Wc,0.05) \quad (1a)$$

On the other hand, in DEFINE STYLE statement on PROC TEMPLATE for the ‘Coding Frame Attributes’ setting on parameters ‘cellpadding’, ‘borderspacing’, and ‘borderwidth’ (see macro %style in APPENDX 2), the coding setting to the width ‘Wc’ has been converted to the actual width ‘Wa’ by rounding the ‘Wc’ to the nearest multiple of 0.05 (unit ‘pt’), and in addition, 0<Wc<0.025 will be rounded to Wa=0.05. In a formula:

$$Wa = round(Wc+(0<Wc<0.025)*0.025,0.05) \quad (1b)$$

Formula (1b) has been verified by macro %Border in APPENDIX 2. Due to space limitations, no further details will be given here.

From formula (1a) and (1b), we know, to keep the setting Coding Column Width 'Wc' the same as Actual Column Width 'Wa' (on value of Column Width, 'cellpadding', 'borderspacing', and 'borderwidth'), if and only if setting 'Wc' is a multiple of 0.05 pt. This rule is under the option 'dpi=1440' on ODS PDF statement.

In a PDF destination, a string's Physical Length 'W' is related to the Actual Column Width 'Wa'. (a) If 'borderspacing=0pt', and 'borderwidth=0pt' are in the style or (b) if 'frame=hsides' is in the style. Whether a string's wrap-up is depends on a formula as:

$$Wa >= W + 2 * cellpadding \text{ (The string without wrap-up)} \quad (2a)$$

$$Wa < W + 2 * cellpadding \text{ (The string with wrap-up)} \quad (2b)$$

In condition: 'cellpadding=0', the formula becomes:

$$Wa >= W \text{ (The string without wrap-up)} \quad (3a)$$

$$Wa < W \text{ (The string with wrap-up)} \quad (3b)$$

So, a minimum value of the column width 'Wa' for a string with Physical length 'W' without wrap-up is called as the **critical actual column width** 'Wa', and in the critical condition Wa=W.

MEASURING PHYSICAL LENGTH IN A PDF FILE WITH 'TIMES NEW ROMAN' FONT

Unfortunately, for many of characters, the Physical Width is not in strict direct proportion to the font size.

The principle of calculating the physical length of a string in 'Times New Roman' is based on following basic rule, which we could call the 'principle of superposition' for either font weight 'Bold', or 'Normal':

> *By removing the trailing blanks (with ASCII code 32), a string's physical length in a specified font size is equal to* ***the cumulative sum of each character's width*** *in the same font size.*

If a string in font size <p>pt is made up of 'n' characters, and each character's width (on font size <p>pt) are given by: '$w_1^{(p)}$, $w_2^{(p)}$, .. , $w_n^{(p)}$', letting W be string's Physical Width, the formula is:

$$W = w_1^{(p)} + w_2^{(p)} + .. + w_n^{(p)} \quad (4)$$

In other words, some of characters with different ASCII codes share the same characteristic width. For example, digits 0, 1, .. , 9 have the same width as long as they are in the same font size. If a string is made up of 'n' characters, 'n_1' characters are with characteristic width '$w_1^{(p)}$', 'n_2' characters are with characteristic width '$w_2^{(p)}$', …, and 'n_k' characters are with characteristic width '$w_k^{(p)}$', where $n= n_1+n_2+_{..}+n_k$. then the formula is:

$$W = n_1 * w_1^{(p)} + n_2 * w_2^{(p)} + .. + n_k * w_k^{(p)} \quad (4a)$$

Each character has 2 characteristic widths, respectively, for both font weight as 'Bold' and 'Normal'. If a string is comprised of 'n' repeated single characters, the formula becomes:

$$W = n \cdot w^{(p)} \quad (4b)$$

Macro %formula in 'APPENDIX 2' prove formula (4b). Due to space limitations, no further details will be given here. A string's physical length 'W' is in strictly proportional to the number of characters 'n' in a string with repeated single character.

The next consideration is the text alignment including left alignment and right alignment. By a macro %Textalign in APPENDIX 2, the output is generated as shown in Figure 5. Here the column S1A, S1B and S1C is the same string 'S1' (300 repeated single quotation mark " ' ") with different column widths (as 535 pt, 525 pt and 524.95 pt). Each string is repeated for both left alignment and right alignment in one cell.

In column width 524.95 pt for S1C, the letters wrap for both strings on left alignment and right alignment, but string S1B is without wrap for both. So, 525 pt (in column as S1B) is the Physical Length of the string with 300 repeated single quotation mark " ' ". For the same reason, 533 pt (in column as S2B) is the Physical Length of the string with 260 repeated single vertical line "|", while 540 pt (in column as S3B) is the Physical Length of the string with 100 repeated tilde "~".

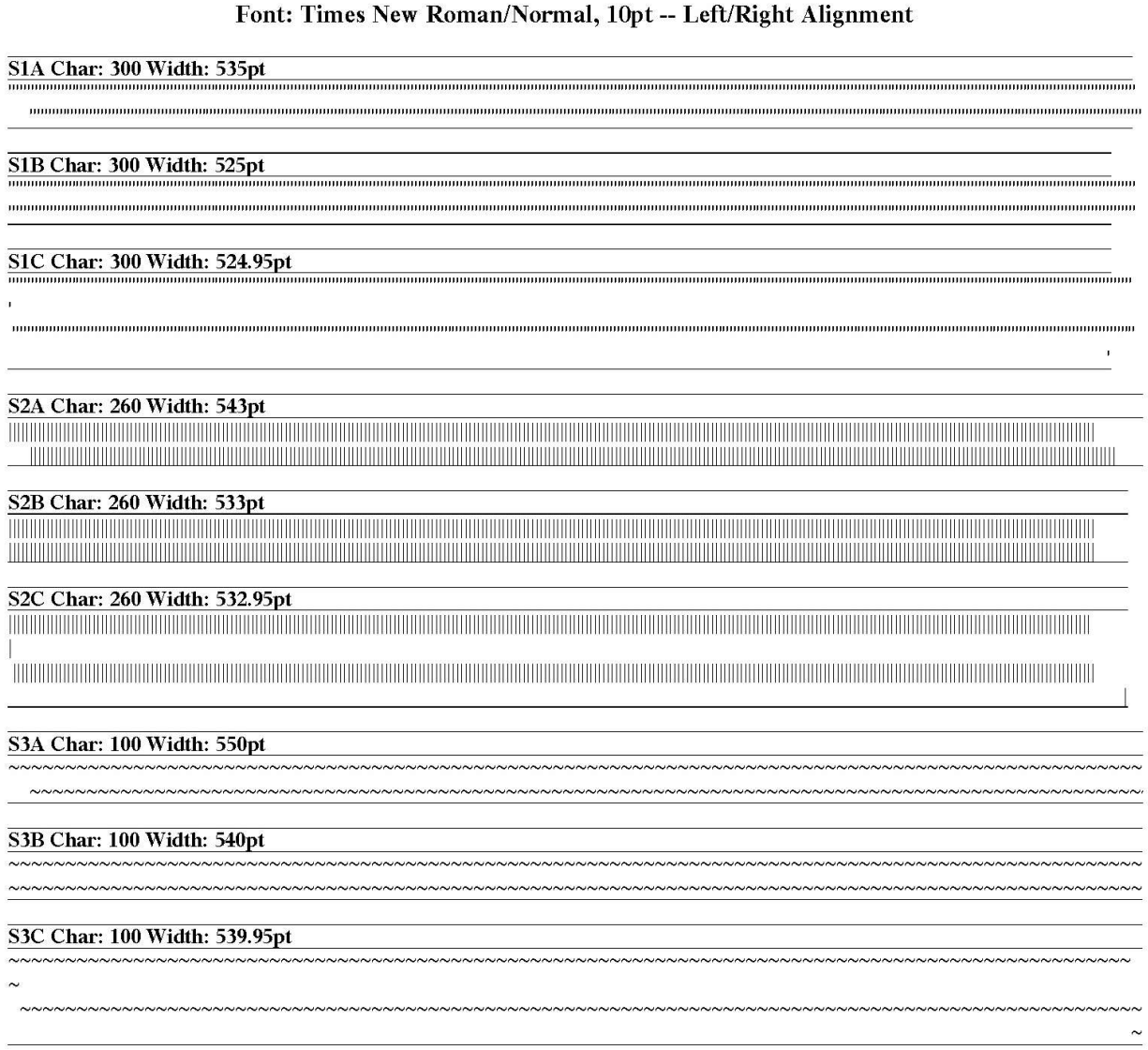

Figure 5: Character " ' ", "|", and "~" respectively left and right aligned in different column widths.

In the Figure 5 on right alignment situation, string S1's Vision Length go over the column's right end/border. And string S2's Vision Length didn't reach to the column's right end/border. For string S1, column S1B set the 525 pt as the correct Physical Length, but its Vision Length is longer than 525 pt. For string S2, column S2B set the 533 pt as the correct Physical Length, but its Vision Length is shorter than 533 pt. For string S3, character '~' have the same Physical Length as the Vision Length (for font weight as 'Normal' and size as 10pt). So, column S3B sets 540 pt as the Physical Length, and string S3's Vision Length is exactly touching the column's right end/border.

For reference, a PDF file is completely generated by ODS PDF with SAS software (version 9.4 [TS1M3)] in this paper). SAS software, however, recognizes only Physical Length for each character without

knowing the Vision Length.

From the analysis as above, while 'cellpadding=0', we reach the following 'principle of text alignment':

> *A string has a fixed physical length regardless of whether left aligned or right aligned. In the left aligned situation, the string's Physical Length (and Vision Length) starting x-coordinate is on the left border of the column.* ***In the right alignment situation, only the string's Physical Length ending x-coordinate is on the right border of the column****.*

Therefore, for a right aligned string, set cellpadding=0pt, borderspacing=0pt, and borderwidth=0pt, we derived formula as following:

<string's physical width> = <string ending x-coordinate> - <string starting x-coordinate>
= <column starting x-coordinate> + <column width> - <string starting x-coordinate> (5)

Formula (5) above could be used to calculate a string's Physical Length, in macro %PhyscalW from 'APPENDIX 2'. Each character's Physical Width from 7pt to 12pt on both 'Bold' or 'Normal' are calculated, and the results are shown in 'APPENDIX 1'.

MEASURING VISION LENGTH IN A PDF FILE WITH 'TIMES NEW ROMAN' FONT

After a PDF file is generated by SAS ODS and saved, an Adobe application is used to open this file (using a version of 'Adobe Acrobat X Pro' in this paper). During its display, Adobe uses string's Vision Length without knowing its Physical Length.

A string's Vision Length may differ from its Physical Length, being either shorter or longer. In addition, a string's Vision Length is in strict direct proportion to the font size. Macro %V_Fontsize verified this fact for font sizes from 1pt to 12pt.

The following is a part of output PDF file 'V_Fontsize2.pdf' by the macro %V_Fontsize. The output in Figure 6 is with 12 lines on 'Normal' font. The rule is: font size (pt) * Number of '!' = 2520. Therefore:

(1) The first line is: 2520 '!' on font size 1pt.

(2) The second line is:1260 '!' on font size 2pt.

..

(11) The eleventh line is: 229 '!' on font size 11pt (2520 can't be divisible by 11; 2520/11=229.0909..).

(12) The twelfth line is: 210 '!' on font size 12pt.

Figure 6: When Font size * Number of Char = Constant, they display with the same length.

To provide more detail, we highlight these strings and output at the right end in Figure 7, Please notice: a string's Vision Length is exactly equal to the length of highlighted band.

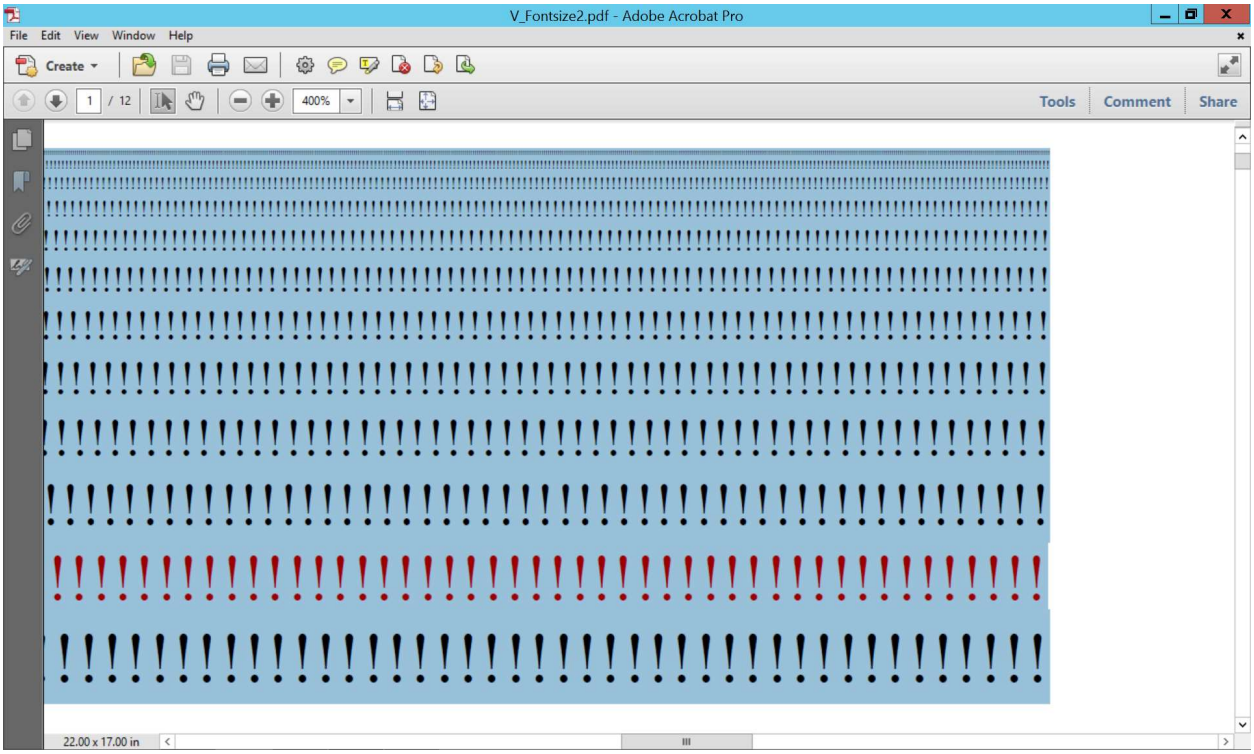

Figure 7: By highlighting, each string (from 1pt to 12pt) has the same length.

The string for 229 '!' on11pt (in red color) is slightly shorter because 2520/11=229.0909, and function floor() is used to get an integer number, 229.

The principle for calculating the Vision Length of a string in font 'Times New Roman' on option 'dpi=1440' is based on the following basic rule, which we could call a 'principle of linear superposition':

> *By removing the trailing blanks (with ASCII code 32), a string's Vision Length is equal to the* ***cumulative sum*** *of each character's vision characteristic width, multiplied by the font size (in pt units).*

If a string is made up of 'n' characters, and each character's vision characteristic width is given by: 'w_1, w_2, .. , w_n', letting $W^{(v)}$ be the string's physical width (in 'pt' units), and 'p' be font size as an integer (in 'pt' units'), then the formula is:

$$W^{(v)} = p*(w_1+w_2+ .. +w_n) \quad (6)$$

In other words, by considering that some characters with different ASCII code share the same Vision Characteristic Width (for example, digit 0, 1, .. , 9 have the same Vision Characteristic Width), if a string consists of 'n' characters, and 'n_1' characters are with Vision Characteristic Width 'w_1', 'n_2' characters are with Vision Characteristic Width 'w_2', …, and 'n_k' characters are with Vision Characteristic Width 'w_k', where $n= n_1+n_2+_{...}+n_k$. then the formula is:

$$W^{(v)} = p*(n_1*w_1+ n_2*w_2+ .. + n_k*w_k) \quad (6a)$$

The measuring characters with ASCII code from 32 to 126 constitute 95 different cases. Each character is with font weight 'Bold' and 'Normal' for measuring individually. While measuring the Vision Characteristic Width for 190 cases, we found that these characters are grouped into 29 groups of values. Macro %Vision_W verified these groups. The following shows the Vision Length in group #10. Four characters 'cerz' in 'Bold' font (with black color in the Figure 8) and five characters '?acez' in 'Normal' font (with red color in the Figure 8) shared the same Vision Length. They are all string with 120 characters.

S10
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
eeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeee
rrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrr
zzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzz
??
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
eeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeee
zzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzz

Figure 8: Vision length group #10. Each line has 120 characters. Black represents 'Bold', red 'Normal'

Because each character group (among 29 groups) share the same value of '*vision characteristic width*', we could only pick up one character from each group to measure the 29 values of '*vision characteristic width*'. The method is comparing a string's vision width with certain length of related cell frame line. The following is a part of output PDF file 'V_Width.pdf' by the macro %V_Width (the code is in 'APPENDIX 2').

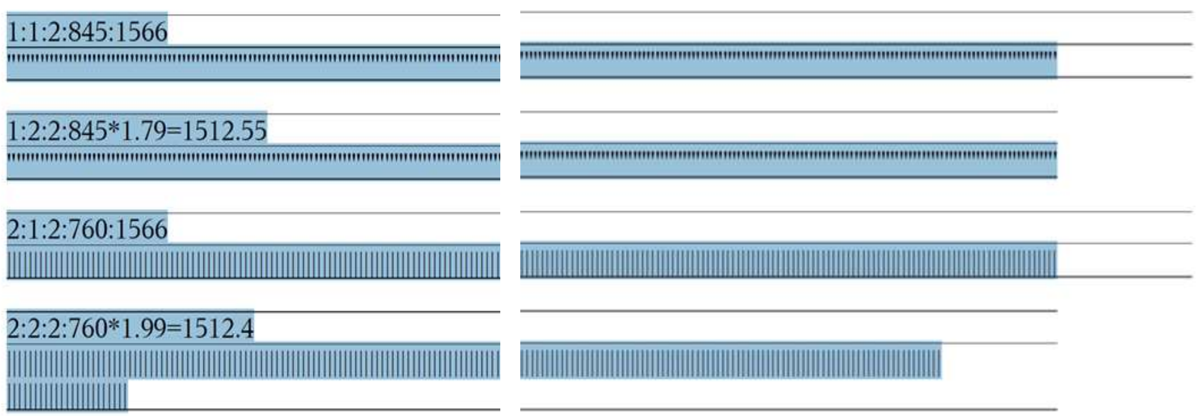

Figure 9: Part of the output on V_Width.pdf, just displyed both enlarged left and right end.

Actually, a string's 'Vision Length' should be measured from the left edge to the right edge on highlighted band. When we watch the output on both sides on row 1 and row 2, both rows have 845 character " ' " (single quote) in font weight 'normal' and size 10pt. In the first row, the 3 horizontal cell frame is in length 1566 pt, and in the second row, the 3 horizontal cell frame is in length 1512.55 pt (by setting SAS code: style=[cellwidth=1512.55pt]). The highlighted band is line-up with the cell frame, now we know 845 continuous single quote (in weight normal and size 10 pt) take 1512.55 pt as the vision length. By formula $W^{(v)} = p*n*w$, and n=845, $W^{(v)}$=1512.55pt, p=10pt, the character " ' " have his 'Vision Characteristic Width': w = 0.179.

Both row 3 and row 4 have 760 character "|". In the row 3, the string is without word wrap up, and in row 4, the 3 horizontal cell frame is in length 1512.4 pt. Though highlighted band in row 4 is wrap up to the second line, the highlighted band in row 3 is line-up with the cell frame in row 4, now we know 760 continuous character "|" take 1512.4 pt as the vision length. By formula $W^{(v)} = p*n*w$, and n=760, $W^{(v)}$=1512.4pt, p=10pt, the letter "|" have his 'Vision Characteristic Width': w = 0.199.

The detail measurements obtained for the 29 groups of Vision Characteristic Width are shown in Figure 10:

Vision Characteristic Width on 'Times New Roman' for Size 1pt to 10 pt

Group	#1	#2	#3	#4	#5	#6	#7	#8	#9	#10	#11	#12	#13	#14
Bold			|	**,(Space).**	**'/\il**	**!()-:;[]`fjt**	**Is**	**{}**		**cerz**			**#$*0123456789?J_agovxy**	**~**
Normal	'	|		,(Space).	/:;\ijlt	!()-I[]`fr	Js		"	?acez	^	{}	#$*0123456789_bdghknopquvxy	
V.C.W.	0.179	0.199	0.219	0.249	0.276	0.332	0.388	0.393	0.407	0.442	0.468	0.479	0.499	0.519

Group	#15	#16	#17	#18	#19	#20	#21	#22	#23	#24	#25	#26	#27	#28	#29
Bold		**"Sbdhknpqu**		**+<=>**	**^**	**FP**	**BELTZ**	**ACDNRUVXYw**	**GHKOQ**	**&m**			**@**	**M**	**%W**
Normal	~	FPS	+<=>			ELTZ	BCR	ADGHKNOQUVXYw	&m	%	M	@		W	
V.C.W.	0.540	0.555	0.563	0.568	0.580	0.609	0.666	0.721	0.776	0.832	0.888	0.919	0.929	0.942	0.999

Figure 10: Vision characteristic width for the 29 groups

THE FUNCTION 'PWIDTH' FOR STRING'S PHYSICAL AND VISION WIDTHS

Next, we considered how to calculate a string's Generalized Physical Length include Physical Length and Vision Length with an SAS user-define function pwidth(). Because the font size from 1pt to 6pt is too small and rarely used, to save coding space in user-define function pwidth(), we only provided the calculations for font sizes from 7pt to 12pt for both font weight 'Bold' or 'Normal' on 'Times New Roman' font. The function is matched with the ODS PDF statement with option 'dpi=1440'.

The function's user interface is pwidth(string, bold, fsize, phy). Parameter 'string' specifies a character string which is for Physical Length or Vision Length calculation. Parameter 'bold' is a number to indicate the font weight. '1' is 'Bold' font weight, '2' as 'Normal' font weight. Parameter 'fsize' is a number from 7 to 12 to indicate the font size in unit 'pt'. Parameter 'phy' is a number to indicate which width is to be calculated. '1' calculates the string's 'Physical Length'; '2' calculates the string's 'Visionl Length', and '3' calculates string's both 'Physical Length' and 'Visionl Length', returning the maximum value. Should a string without wrap up to the second line and without overlapping to the next column be desired, choose 'phy=3'.

```
proc fcmp outlib=work.funcs.maths;
  function pwidth(string $, bold, fsize, phy);
          array c[47] $20 (
'|' ' ,.' ''''' '/\il' '!' ':;t' 'j' ']' 'f' '()-`' '[' 'I' 's' '{}' 'r' 'z' 'ce' 'J' 'a' '?' '#$*0123456789_gov' 'x' 'y' '~'
'k' '"' 'bdhnpqu' 'S' '+<=>' '^' 'FP' 'Z' 'T' 'EL' 'B' 'CR' 'ADNUY' 'V' 'w' 'X' 'GHKOQ' 'm' '&' '@' 'M' '%' 'W');

          array w[6,2,47] (
1.45 1.75 1.95 1.95 2.30 2.35 2.35 2.35 2.35 2.35 2.35 2.70 2.70 2.75 3.10 3.10 3.10 3.50 3.50 3.50 3.50 3.50 3.55 3.65
3.80 3.90 3.90 3.90 4.00 4.05 4.30 4.55 4.65 4.65 4.65 5.05 5.05 5.10 5.10 5.15 5.45 5.70 5.85 6.50 6.60 7.00 7.00
1.35 1.75 1.25 1.95 2.35 1.95 1.95 2.30 2.30 2.35 2.35 2.35 2.70 3.35 2.35 3.10 3.10 2.70 3.10 3.20 3.50 3.50 3.55 3.80
3.50 2.85 3.50 3.90 3.95 3.30 3.90 4.30 4.25 4.30 4.65 4.65 5.05 5.05 5.05 5.05 5.05 5.35 5.40 6.45 6.20 5.85 6.60
1.75 2.00 2.20 2.20 2.65 2.65 2.65 2.65 2.65 2.65 3.10 3.10 3.15 3.55 3.45 3.55 4.00 4.00 4.00 4.00 4.05 4.05 4.15
4.45 4.45 4.45 4.45 4.55 4.65 4.90 5.25 5.35 5.35 5.35 5.80 5.80 5.75 5.90 5.85 6.20 6.75 6.65 7.45 7.55 8.15 8.05
1.55 2.00 1.45 2.20 2.65 2.20 2.25 2.65 2.65 2.65 2.70 2.65 3.10 3.85 2.65 3.55 3.55 3.10 3.55 3.65 4.00 4.00 4.10 4.35
4.00 3.25 4.00 4.45 4.50 3.75 4.45 4.90 4.85 4.90 5.35 5.35 5.80 5.80 5.80 5.80 5.80 6.15 6.15 7.35 7.10 6.65 7.60
1.90 2.25 2.50 2.50 3.00 3.00 3.00 3.00 3.05 3.00 3.00 3.50 3.50 3.55 4.00 3.90 4.00 4.50 4.50 4.50 4.50 4.55 4.60 4.70
4.95 5.00 5.00 5.00 5.15 5.25 5.50 5.90 6.00 6.00 6.00 6.50 6.50 6.45 6.55 6.55 7.00 7.55 7.50 8.35 8.50 8.95 9.00
1.80 2.25 1.60 2.50 3.00 2.50 2.55 3.05 3.00 3.00 3.05 3.00 3.50 4.30 3.00 4.00 4.00 3.50 4.00 4.10 4.50 4.50 4.60 4.85
4.50 3.65 4.50 5.00 5.10 4.20 5.00 5.50 5.55 5.50 6.00 6.00 6.50 6.50 6.50 6.50 6.50 6.95 7.00 8.30 8.00 7.50 8.50
2.15 2.50 2.80 2.80 3.30 3.35 3.35 3.35 3.35 3.35 3.35 3.90 3.90 3.95 4.45 4.35 4.45 5.00 5.00 5.00 5.00 5.00 5.20
5.60 5.55 5.55 5.55 5.70 5.80 6.10 6.55 6.65 6.65 6.65 7.20 7.20 7.15 7.15 7.25 7.80 8.35 8.35 9.30 9.45 9.90 10.00
2.05 2.50 1.75 2.80 3.35 2.80 2.80 3.30 3.30 3.35 3.30 3.35 3.90 4.80 3.35 4.45 4.45 3.90 4.45 4.50 5.00 5.00 5.10 5.40
5.00 4.10 5.00 5.55 5.65 4.70 5.55 6.10 6.15 6.10 6.65 6.65 7.20 7.20 7.20 7.20 7.70 7.75 9.20 8.90 8.35 9.35
2.35 2.75 3.05 3.05 3.65 3.65 3.65 3.65 3.65 3.65 3.65 4.30 4.30 4.35 4.90 4.85 4.90 5.50 5.50 5.50 5.50 5.55 5.50 5.70
6.00 6.10 6.10 6.10 6.25 6.40 6.70 7.25 7.35 7.35 7.35 7.95 7.95 7.90 7.95 7.95 8.55 9.20 9.15 10.25 10.40 11.05 11.00
2.20 2.75 1.90 3.05 3.65 3.05 3.05 3.55 3.65 3.65 3.60 3.65 4.30 5.30 3.65 4.90 4.90 4.30 4.90 5.05 5.50 5.50 5.65 5.95
5.50 4.50 5.50 6.10 6.20 5.15 6.10 6.70 6.75 6.70 7.35 7.35 7.95 7.95 7.95 7.95 7.95 8.55 8.50 10.15 9.80 9.15 10.35
2.60 3.00 3.35 3.35 3.95 4.00 4.00 4.00 4.00 4.00 4.00 4.65 4.65 4.75 5.35 5.30 5.35 6.00 6.00 6.00 6.00 6.00 6.25
6.65 6.65 6.65 6.65 6.85 6.95 7.35 7.85 8.00 8.00 8.00 8.65 8.65 8.65 8.55 8.75 9.35 10.05 10.00 11.15 11.35 11.90 12.00
2.45 3.00 2.10 3.35 4.00 3.35 3.30 3.95 3.95 4.00 3.95 4.00 4.65 5.75 4.00 5.35 5.35 4.65 5.35 5.40 6.00 6.00 6.10 6.50
6.00 4.90 6.00 6.65 6.75 5.65 6.65 7.35 7.35 7.35 8.00 8.00 8.65 8.65 8.65 8.65 8.65 9.20 9.35 11.05 10.65 10.00 11.30);
          array d[2,29] $30 (
' ' ' ' '|' ', .' '''/\il' '!()-:;[]`fjt' 'Is' '{}' ' ' 'cerz' ' ' ' ' '#$*0123456789?J_agovxy'
'~' ' ' '"Sbdhknpqu' ' ' '+<=>' '^' 'FP' 'BELTZ' 'ACDNRUVXYw' 'GHKOQ' '&m' ' ' ' ' ' '@' 'M' '%W'
''''' '|' ' ' ', .' '/:;\ijlt' '!()-I[]`fr' 'Js' ' ' '"' '?acez' '^' '{}' '#$*0123456789_bdghknopquvxy'
' ' '~' 'FPS' '+<=>' ' ' ' ' 'ELTZ' 'BCR' 'ADGHKNOQUVXYw' '&m' '%' 'M' '@' ' ' 'W' ' ');
          array v[29]   (1.79 1.99 2.19 2.49 2.76 3.32 3.88 3.93 4.07 4.42 4.68 4.79 4.99
          5.19 5.40 5.55 5.63 5.68 5.80 6.09 6.66 7.21 7.76 8.32 8.88 9.19 9.29 9.42 9.99);
          array WD[3] WD1-WD3;
    length string $1250;
    if phy^=2 then do i=1 to 47;
                  WD1 + countc(string,c[i],'T')*w[fsize-6,bold,i];
    end;
    if phy^=1 then do;
```

```
                do i=1 to 29; WD2 + countc(string,d[bold,i],'T')*v[i];
                end;
                WD2 = WD2*fsize/10;
                if phy=3 then WD3 = max(WD1,WD2);
    end;
    return(WD[phy]);
  endfunc;
run; quit;
options cmplib=work.funcs;
```

CODING EXAMPLE

Here, we provide a SAS code as an example to show how to decide on the column's width setting to display a dataset while using PROC REPORT with ODS RTF. The code is macro %CLNMSG2 in 'APPENDIX 2'. The input dataset is 'sashelp.CLNMSG', but only the first 4 observations are used/displayed, and with a suitable column width without word wrap up. In macro %CLNMSG2, macro %style is invoked as follows:

```
%style(stylen=6, boldn=2, frame=1, rules=2, fontsize=10, margin=50,
       cellpadding=4, borderspacing=0.75, borderwidth=0.75);
```

To choose a suitable column width, we should consider the physical width of both the table header and contents. So 'PROC CONTENTS' is used for getting the variable's name and label in a dataset as variables 'NAME' and 'LABEL'. The following piece of code in a 'PROC SQL' calculated the header's texts on the Physical Width:

```
pwidth(coalescec(LABEL,NAME),1,10,1)
```

Function 'coalescec()' is used in case of 'LABEL' is missing, use 'NAME' in place of the header. The second argument in pwidth() uses '1' -- for 'Bold' font in the header part. The third argument in pwidth() is as '10' -- for font size in 10pt on header. And the fourth argument in pwidth() is as '1' -- to calculate Physical Length only. When not calculating 'vision length', no wrap up happened, but sometimes the text could overlap into the next column. In the next step, while a variable' name is assigned to macro variable '&&VR&i' and the header's Physical Length has been assigned as '&&LBL&i', the following code decided the column's width:

```
max(&&LBL&i,max(pwidth(cat(&&VR&i),2,10,1)))+2*4
```

Code 'pwidth(cat(&&VR&i),2,10,1)' get each observation's Physical Length, while max() get the maximum value of all observations in the 'PROC SQL'. Code 'max(&&LBL&i,max(pwidth(cat(&&VR&i),2,10,1)))' get the maximum value for both 'table header' and 'table content' parts. Then code piece '+2*4' is for reason of formula (2a) in this paper (add double of 'cellpadding').

(13) Set Column Width by function pwidth()

MSGID	MNEMONIC	LINENO	LEVEL	TEXT	PBUTTONS
14	IN_SEL_NOT_AVAIL	1	N	The selection %1$ is not available.	SASHELP.FSP.OK,SLIST
24	IN_NOT_AVAILABLE	1	N	This option is not yet available.	SASHELP.FSP.OK,SLIST
32	IO_DICT_CANNOT_OPEN_STUDY	1	E	Unable to open the study. There are problems with the dictionary.	SASHELP.FSP.OK,SLIST
82	IO_NO_ACCESS	1	E	%1You do not have authorization to access %1$.	SASHELP.FSP.OK,SLIST

MSGID	MNEMONIC	LINENO	LEVEL	TEXT	PBUTTONS
14	IN_SEL_NOT_AVAIL	1	N	The selection %1$ is not available.	SASHELP.FSP.OK. SLIST
24	IN_NOT_AVAILABLE	1	N	This option is not yet available.	SASHELP.FSP.OK. SLIST
32	IO_DICT_CANNOT_OPEN_ STUDY	1	E	Unable to open the study. There are problems with the dictionary.	SASHELP.FSP.OK. SLIST
82	IO_NO_ACCESS	1	E	%1You do not have authorization to access %1S.	SASHELP.FSP.OK. SLIST

Figure 11: The table in the top used function 'pwidth()' and the formula (2) for the column setting in PROC REPORT. The table in the bottom is minus only 0.05pt for each column and wraps either on table contents, or on table header.

As shown in Figure 11, the dataset CLNMSG is displayed twice in nearly the same way, but the table in the bottom is 0.05pt less on each column's width. As a results, the top table is without any word/letter wrap up, but the bottom table is with word/letter wrap up either on the header or in table contents for each column with this small change. Figure 12 shows the SAS log file for the column width settting for each variable's column.

```
MPRINT(CLNMSG2):   ods listing close;
MPRINT(CLNMSG2):   ods pdf file="C:\Users\william.wu\program\paper\file\CLNMSG.pdf" style=Mystyle6_2 startpage=no dpi=1440 notoc compress=0;
NOTE: Writing ODS PDF output to DISK destination "C:\Users\william.wu\program\paper\file\CLNMSG.pdf", printer "PDF".
MPRINT(CLNMSG2):   title j=c "(13) Set Column Width by function pwidth()";
MPRINT(CLNMSG2):   proc report data=CLNMSG nowd;
MPRINT(CLNMSG2):   column MSGID MNEMONIC LINENO LEVEL TEXT PBUTTONS;
MPRINT(CLNMSG2):   define MSGID /display style=[just=c cellwidth=41.9pt];
MPRINT(CLNMSG2):   define MNEMONIC /display style=[just=l cellwidth=162.85pt];
MPRINT(CLNMSG2):   define LINENO /display style=[just=c cellwidth=47.4pt];
MPRINT(CLNMSG2):   define LEVEL /display style=[just=c cellwidth=41.75pt];
MPRINT(CLNMSG2):   define TEXT /display style=[just=l cellwidth=271.85pt];
MPRINT(CLNMSG2):   define PBUTTONS /display style=[just=l cellwidth=116.5pt];
MPRINT(CLNMSG2):   run;

NOTE: There were 4 observations read from the data set WORK.CLNMSG.
NOTE: PROCEDURE REPORT used (Total process time):
      real time           0.03 seconds
      cpu time            0.03 seconds


MPRINT(CLNMSG2):   proc report data=CLNMSG nowd;
MPRINT(CLNMSG2):   column MSGID MNEMONIC LINENO LEVEL TEXT PBUTTONS;
MPRINT(CLNMSG2):   define MSGID /display style=[just=c cellwidth=41.85pt];
MPRINT(CLNMSG2):   define MNEMONIC /display style=[just=l cellwidth=162.8pt];
MPRINT(CLNMSG2):   define LINENO /display style=[just=c cellwidth=47.35pt];
MPRINT(CLNMSG2):   define LEVEL /display style=[just=c cellwidth=41.7pt];
MPRINT(CLNMSG2):   define TEXT /display style=[just=l cellwidth=271.8pt];
MPRINT(CLNMSG2):   define PBUTTONS /display style=[just=l cellwidth=116.45pt];
MPRINT(CLNMSG2):   run;

NOTE: There were 4 observations read from the data set WORK.CLNMSG.
NOTE: PROCEDURE REPORT used (Total process time):
      real time           0.01 seconds
      cpu time            0.01 seconds
```

Figure 12: The code the SAS log showing the width setting on each column

CONCLUSION

This paper presents some basic methods and rules to solve Physical Length and Vision Length calculations for font 'Times New Roman' (or font 'Times Roman') on ODS PDF destination. The paper proved the 'principle of superposition' for Physical Length, and 'principle of linear superposition' for Vision Length. On option 'dpi=1440' in ODS PDF statement, we obtained a custom SAS function 'pwidth()'.

Using a similar method, we could calculate Physical Length and Vision Length in PDF file for fonts other than 'Times New Roman'.

ACKNOWLEDGMENTS

We thank Robert Rydzewski, MS, for help with copyediting of this paper.

REFERENCES

[1] William Wu, and Steven Li (2023), "Calculate Physical Length of String in RTF File with 'Times New Roman' by SAS", Proceeding of Southeast SAS Users Group 2023, Paper 105. Available at

https://www.lexjansen.com/sesug/2023/SESUG2023_Paper_105_Final_PDF.pdf

[2] William Wu, Steve Li, and Yun Guan (2016), "Importing Data Directly from PDF into SAS® Data Sets", SAS Global Forum 2016 Proceedings, Session 9320-2016. Available at
https://support.sas.com/resources/papers/proceedings16/9320-2016.pdf

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

William Wu

Astex Pharmaceuticals, Inc.

Phone: (650)350-8604

Email: willywu2001@hotmail.com

Steven Li

Medtronic PLC., Mounds View, MN

Email: steven_li99@hotmail.com

APPENDIX 1

A Character's Physical Width (Unit: pt) for Both Normal and Bold From 7pt To 12pt

	Size: 7pt		Size: 8pt		Size: 9pt		Size: 10pt		Size: 11pt		Size: 12pt	
Characters	Bold	Norm	Bold	Norm	Bold	Norm	Bold	Norm	Bold	Norm	Bold	Norm
|	1.45	1.35	1.75	1.55	1.90	1.80	2.15	2.05	2.35	2.20	2.60	2.45
(Space),.	1.75	1.75	2.00	2.00	2.25	2.25	2.50	2.50	2.75	2.75	3.00	3.00
'	1.95	1.25	2.20	1.45	2.50	1.60	2.80	1.75	3.05	1.90	3.35	2.10
Âìl	1.95	1.95	2.20	2.20	2.50	2.50	2.80	2.80	3.05	3.05	3.35	3.35
!	2.30	2.35	2.65	2.65	3.00	3.00	3.30	3.35	3.65	3.65	3.95	4.00
:;t	2.35	1.95	2.65	2.20	3.00	2.50	3.35	2.80	3.65	3.05	4.00	3.35
j	2.35	1.95	2.65	2.25	3.00	2.55	3.35	2.80	3.65	3.05	4.00	3.30
]	2.35	2.30	2.65	2.65	3.00	3.05	3.35	3.30	3.65	3.55	4.00	3.95
f	2.35	2.30	2.65	2.65	3.05	3.00	3.35	3.30	3.65	3.65	4.00	3.95
()-`	2.35	2.35	2.65	2.65	3.00	3.00	3.35	3.35	3.65	3.65	4.00	4.00
[	2.35	2.35	2.65	2.70	3.00	3.05	3.35	3.30	3.65	3.60	4.00	3.95
I	2.70	2.35	3.10	2.65	3.50	3.00	3.90	3.35	4.30	3.65	4.65	4.00
s	2.70	2.70	3.10	3.10	3.50	3.50	3.90	3.90	4.30	4.30	4.65	4.65
{}	2.75	3.35	3.15	3.85	3.55	4.30	3.95	4.80	4.35	5.30	4.75	5.75
r	3.10	2.35	3.55	2.65	4.00	3.00	4.45	3.35	4.90	3.65	5.35	4.00
z	3.10	3.10	3.45	3.55	3.90	4.00	4.35	4.45	4.85	4.90	5.30	5.35
ce	3.10	3.10	3.55	3.55	4.00	4.00	4.45	4.45	4.90	4.90	5.35	5.35
J	3.50	2.70	4.00	3.10	4.50	3.50	5.00	3.90	5.50	4.30	6.00	4.65
a	3.50	3.10	4.00	3.55	4.50	4.00	5.00	4.45	5.50	4.90	6.00	5.35
?	3.50	3.20	4.00	3.65	4.50	4.10	5.00	4.50	5.50	5.05	6.00	5.40
#$*0123456789_gov	3.50	3.50	4.00	4.00	4.50	4.50	5.00	5.00	5.50	5.50	6.00	6.00
x	3.50	3.50	4.05	4.00	4.55	4.50	5.00	5.00	5.55	5.50	6.00	6.00
y	3.55	3.55	4.05	4.10	4.60	4.60	5.00	5.10	5.50	5.65	6.00	6.10
~	3.65	3.80	4.15	4.35	4.70	4.85	5.20	5.40	5.70	5.95	6.25	6.50
k	3.80	3.50	4.45	4.00	4.95	4.50	5.60	5.00	6.00	5.50	6.65	6.00
"	3.90	2.85	4.45	3.25	5.00	3.65	5.55	4.10	6.10	4.50	6.65	4.90
bdhnpqu	3.90	3.50	4.45	4.00	5.00	4.50	5.55	5.00	6.10	5.50	6.65	6.00
S	3.90	3.90	4.45	4.45	5.00	5.00	5.55	5.55	6.10	6.10	6.65	6.65
+<=>	4.00	3.95	4.55	4.50	5.15	5.10	5.70	5.65	6.25	6.20	6.85	6.75
^	4.05	3.30	4.65	3.75	5.25	4.20	5.80	4.70	6.40	5.15	6.95	5.65
FP	4.30	3.90	4.90	4.45	5.50	5.00	6.10	5.55	6.70	6.10	7.35	6.65
Z	4.55	4.30	5.25	4.90	5.90	5.50	6.55	6.10	7.25	6.70	7.85	7.35
T	4.65	4.25	5.35	4.85	6.00	5.55	6.65	6.15	7.35	6.75	8.00	7.35
EL	4.65	4.30	5.35	4.90	6.00	5.50	6.65	6.10	7.35	6.70	8.00	7.35
B	4.65	4.65	5.35	5.35	6.00	6.00	6.65	6.65	7.35	7.35	8.00	8.00
CR	5.05	4.65	5.80	5.35	6.50	6.00	7.20	6.65	7.95	7.35	8.65	8.00
ADNUY	5.05	5.05	5.80	5.80	6.50	6.50	7.20	7.20	7.95	7.95	8.65	8.65
V	5.10	5.05	5.75	5.80	6.45	6.50	7.15	7.20	7.90	7.95	8.65	8.65
w	5.10	5.05	5.90	5.80	6.55	6.50	7.15	7.20	7.95	7.95	8.55	8.65
X	5.15	5.05	5.85	5.80	6.55	6.50	7.25	7.20	7.95	7.95	8.75	8.65
GHKOQ	5.45	5.05	6.20	5.80	7.00	6.50	7.80	7.20	8.55	7.95	9.35	8.65
m	5.70	5.35	6.75	6.15	7.55	6.95	8.35	7.70	9.20	8.55	10.05	9.20
&	5.85	5.40	6.65	6.15	7.50	7.00	8.35	7.75	9.15	8.50	10.00	9.35
@	6.50	6.45	7.45	7.35	8.35	8.30	9.30	9.20	10.25	10.15	11.15	11.05
M	6.60	6.20	7.55	7.10	8.50	8.00	9.45	8.90	10.40	9.80	11.35	10.65
%	7.00	5.85	8.15	6.65	8.95	7.50	9.90	8.35	11.05	9.15	11.90	10.00
W	7.00	6.60	8.05	7.60	9.00	8.50	10.00	9.35	11.00	10.35	12.00	11.30

APPENDIX 2

This part provided the code for custom function ‘pwidth()’, and the code for 11 macros. The macros are:

1. Macro %style: This macro using PROC TEMPLATE generates various styles for use by the other macro on ODS PDF destination. The style is with font name as “Times New Roman”. The macro allows users to adjust parameters such as ‘fontsize’, ‘cellpadding’, ‘borderspacing’, and ‘borderwidth’.
2. Macro %location: This macro reads the PDF file and finds text and its (x,y) coordinates. It is useful for locating each string by the other macro.
3. Macro %Char_y_m: Shown in Figure 1 in the paper. This output illustrats the difference between Physical Length and Vision Length.
4. Macro %Wc_Wa: The outputs are shown in Figure 2 and 3 in the paper. It shows that ‘Wc’ (‘Coding Column Width’) and Wa (‘Actual Column Width’) can differ. The relationship is followed by formula (1a).
5. Macro %Border: While invoking as %Border(Test=1), it shows that ‘Wc’ (‘Coding Width’) and Wa (‘Actual Width’) are slightly different sometimes on ‘borderspacing’ and ‘borderwidth’. While invoking as %Border(Test=2), it shows that ‘Wc’ (‘Coding Width’) and Wa (‘Actual Width’) are slightly different sometimes on ‘cellpadding’. The relationship is followed by formula (1b).
6. Macro %Textalign: Compared different character’s Physical Length on both ‘left’ and ‘right’ alignment. Helps to get ‘principle of text alignment’ and related formula (5).
7. Macro %Formula: Verified formula (4b). - Physical Length ‘W’ is directly proportional to the number of characters ‘n’ in a string with a continuous single character.
8. Macro %PhysicalW: Verified 95 different character’s physical length on both ‘Bold’ and ‘Normal’ font. The SAS custom function pwidth() is generated based on the data from this macro. A table with physical length data are generated as shown in APPEXDIX 1.
9. Macro %V_Fontsize: demonstrated a string’s Vision Length is in strict direct proportion to the font size.
10. Macro %Vision_W: Verified some of character shared the same Vision Width, and that all characters' Vision Width could be grouped into 29 groups by the values (shown in Figure 9).
11. Macro %V_Width calculated the Vision characteristic width for the 29 groups. So, Figure 10 gets verified by this macro.
12. Macro %CLNMSG2 is an example code demonstrating how to code for deciding on the Column Length setting for each column on PROC REPORT. The output file is ‘CLNMSG.rtf’.

```
%let path = <Set your location for PDF file and SAS data set>;

libname path "&path";

options mprint nodate nonumber nocenter nobyline ls=128 varlenchk=nowarn noquotelenmax missing=' ';
ods path reset;
ods path (prepend) work.templat(update) sasuser.templat(read) sashelp.tmplmst(read);
ods escapechar=' ';
ods _all_ close;
ods listing;

%macro style(stylen=, boldn=2, frame=1, rules=2, fontsize=10, margin=45
           , cellpadding=3, borderspacing=0.5, borderwidth=1);

%local x FT;
%let FT = Title Title Strong Emphasis FixedEmphasis FixedStrong FixedHeading BatchFixed
Fixed HeadingEmphasis Heading Doc;

proc template;
  define style Mystyle&stylen._&boldn /store=work.templat;
    parent=styles.Printer;
    replace fonts /%do x=1 %to 12; "%scan(&FT,&x)Font%scan(2,&x)"=
      ("Times New Roman",%scan(&fontsize %eval(10+2*(&x=2)-(&x=9)),2-(&x>10),' ')pt
        %if %index(2367110,&x)>0 | &boldn&x=112 %then, Bold;)
    %end;;
    replace color_list /'link'=blue 'bgH'=white 'fg'=black 'bg'=white;
    style body from body /leftmargin=&margin.pt rightmargin=&margin.pt topmargin=&margin.pt
        bottommargin=&margin.pt;
    style SystemFooter from SystemFooter /just=left protectspecialchars=on font=Fonts('FixedFont');
    style table from table /frame=%scan(hsides box,&frame) rules=%scan(all groups,&rules)
        cellpadding=&cellpadding.pt borderspacing=&borderspacing.pt borderwidth=&borderwidth.pt;
  end;
run;
```

```
%mend style;

%macro Location(fname=);

%local i _t_1 _t_2 _t_3 _rc_;

%let _t_1 = \d+\.?\d*;
%let _t_2 = ( (&_t_1) &_t_1 &_t_1 &_t_1 re B\*( \[\]0 d){0,2})?( &_t_1 &_t_1
&_t_1 rg &_t_1 &_t_1 &_t_1 RG( &_t_1 w( \[\]0 d)? \d J \d j( \[\]0 d)?)?)?;
%let _t_3 = &_t_2&_t_2&_t_2 BT\s?\/(TT|F)\d &_t_1 Tf (&_t_1) (&_t_1) Td
\(([^\(\)]*(\([^\(\)]*\)[^\(\)]*)*)\)Tj ET;
%let _rc_ = 35;

data _tmp_1(keep=string page);
          length _S_0 $32767 string $8800;
          array _S_[&_rc_] $25000;
          retain _O_ 0 re page;
          if _N_=1 then re = prxparse("/&_t_3/i");
          infile "&fname..pdf" truncover lrecl=%eval(25000*&_rc_) end=eof;
          input (_S_1-_S_&_rc_) ($char25000.);
          _i_ = 1;
          _p0_ = 7768;
          substr(_S_0,_p0_) = _S_[_i_];
          _p1_ = 1;
          do until(_i_=&_rc_ & _p1_=0);
                    if _i_<&_rc_ & (_p1_=0 | _p0_>25000) then do;
                              _i_ + 1;
                              _p0_ = ifn(_p1_=0,1,_p0_-25000);
                              substr(_S_0,_p0_) = substr(_S_0,_p0_+25000)||_S_[_i_];
                    end;
                    call prxnext(re,_p0_,-1,_S_0,_p1_,_L1_);
                    if _p1_>0 then do;
                              if _N_>_O_ then page + 1;
                              string = substr(_S_0,_p1_,_L1_);
                              if _i_=&_rc_ then put 'WAR' "NING: _rc_ = &_rc_  " _i_=;
                              output;
                              _O_ = _N_;
                    end;
          end;
          if eof then call symputx('_O_',_O_,'L');
run;

data _Location_(keep=K: Loc1 Loc2 Text len);
          array _R_[3] _temporary_;
          if _N_=1 then do i=1 to 3;
                    _R_[i] = prxparse(cat("s/&_t_3/$",choosen(i,23,24,25),"/"));
          end;
          set _tmp_1;
          by page;
          length Text $8800;
          array Loc[3];
          array K[5] K1-K5;
          retain K1-K5;
          do i=1 to 2;
                    Loc[i] = input(prxchange(_R_[i],-1,string),best.);
          end;
          Text = prxchange(_R_[3],-1,string);
          call prxposn(_R_[3],25,_Ps_,len);
          if verify(Text,': 0.123456789')=0 & index(Text,':')>1 then do i=1 to 5;
                    K[i] = input(scan(Text,i,':'),best12.);
          end;
run;

%mend Location;

proc fcmp outlib=work.funcs.maths;
  function pwidth(string $, bold, fsize, phy);
          array c[47] $20 (
'|' ' ,.' '''' '/\il' '!' ':;t' 'j' ']' 'f' '()-`' '[' 'I' 's' '{}' 'r' 'z' 'ce' 'J' 'a' '?' '#$*0123456789_gov' 'x' 'y' '~'
'k' '"' 'bdhnpqu' 'S' '+<=>' '^' 'FP' 'Z' 'T' 'EL' 'B' 'CR' 'ADNUY' 'V' 'w' 'X' 'GHKOQ' 'm' '&' '@' 'M' '%' 'W');
          array w[6,2,47] (
1.45 1.75 1.95 1.95 2.30 2.35 2.35 2.35 2.35 2.35 2.35 2.70 2.70 2.75 3.10 3.10 3.10 3.50 3.50 3.50 3.50 3.50 3.55 3.65
3.80 3.90 3.90 3.90 4.00 4.05 4.30 4.55 4.65 4.65 4.65 5.05 5.05 5.10 5.10 5.15 5.45 5.70 5.85 6.50 6.60 7.00 7.00
1.35 1.75 1.25 1.95 2.35 1.95 1.95 2.30 2.30 2.35 2.35 2.35 2.70 3.35 2.35 3.10 3.10 2.70 3.10 3.20 3.50 3.50 3.55 3.80
3.50 2.85 3.50 3.90 3.95 3.30 3.90 4.30 4.25 4.30 4.65 4.65 5.05 5.05 5.05 5.05 5.05 5.35 5.40 6.45 6.20 5.85 6.60
1.75 2.00 2.20 2.20 2.65 2.65 2.65 2.65 2.65 2.65 2.65 3.10 3.10 3.15 3.55 3.45 3.55 4.00 4.00 4.00 4.00 4.05 4.05 4.15
4.45 4.45 4.45 4.45 4.55 4.65 4.90 5.25 5.35 5.35 5.35 5.80 5.80 5.75 5.90 5.85 6.20 6.75 6.65 7.45 7.55 8.15 8.05
1.55 2.00 1.45 2.20 2.65 2.20 2.25 2.65 2.65 2.65 2.70 2.65 3.10 3.85 2.65 3.55 3.55 3.10 3.55 3.65 4.00 4.00 4.10 4.35
4.00 3.25 4.00 4.45 4.50 3.75 4.45 4.90 4.85 4.90 5.35 5.35 5.80 5.80 5.80 5.80 5.80 6.15 6.15 7.35 7.10 6.65 7.60
1.90 2.25 2.50 2.50 3.00 3.00 3.00 3.00 3.05 3.00 3.00 3.50 3.50 3.55 4.00 3.90 4.00 4.50 4.50 4.50 4.50 4.55 4.60 4.70
4.95 5.00 5.00 5.00 5.15 5.25 5.50 5.90 6.00 6.00 6.00 6.50 6.50 6.45 6.55 6.55 7.00 7.55 7.50 8.35 8.50 8.95 9.00
1.80 2.25 1.60 2.50 3.00 2.50 2.55 3.05 3.00 3.00 3.05 3.00 3.50 4.30 3.00 4.00 4.00 3.50 4.00 4.10 4.50 4.50 4.60 4.85
4.50 3.65 4.50 5.00 5.10 4.20 5.00 5.50 5.55 5.50 6.00 6.00 6.50 6.50 6.50 6.50 6.50 6.95 7.00 8.30 8.00 7.50 8.50
2.15 2.50 2.80 2.80 3.30 3.35 3.35 3.35 3.35 3.35 3.35 3.90 3.90 3.95 4.45 4.35 4.45 5.00 5.00 5.00 5.00 5.00 5.00 5.20
5.60 5.55 5.55 5.55 5.70 5.80 6.10 6.55 6.65 6.65 6.65 7.20 7.20 7.15 7.15 7.25 7.80 8.35 8.35 9.30 9.45 9.90 10.00
2.05 2.50 1.75 2.80 3.35 2.80 2.80 3.30 3.30 3.35 3.30 3.35 3.90 4.80 3.35 4.45 4.45 3.90 4.45 4.50 5.00 5.00 5.10 5.40
5.00 4.10 5.00 5.55 5.65 4.70 5.55 6.10 6.15 6.10 6.65 6.65 7.20 7.20 7.20 7.20 7.20 7.70 7.75 9.20 8.90 8.35 9.35
2.35 2.75 3.05 3.05 3.65 3.65 3.65 3.65 3.65 3.65 3.65 4.30 4.30 4.35 4.90 4.85 4.90 5.50 5.50 5.50 5.55 5.50 5.70
6.00 6.10 6.10 6.10 6.25 6.40 6.70 7.25 7.35 7.35 7.35 7.95 7.95 7.90 7.95 7.95 8.55 9.20 9.15 10.25 10.40 11.05 11.00
2.20 2.75 1.90 3.05 3.65 3.05 3.05 3.55 3.65 3.65 3.60 3.65 4.30 5.30 3.65 4.90 4.90 4.30 4.90 5.05 5.50 5.50 5.65 5.95
5.50 4.50 5.50 6.10 6.20 5.15 6.10 6.70 6.75 6.70 7.35 7.35 7.95 7.95 7.95 7.95 8.55 8.50 10.15 9.80 9.15 10.35
2.60 3.00 3.35 3.35 3.95 4.00 4.00 4.00 4.00 4.00 4.00 4.65 4.65 4.75 5.35 5.30 5.35 6.00 6.00 6.00 6.00 6.00 6.00 6.25
6.65 6.65 6.65 6.65 6.85 6.95 7.35 7.85 8.00 8.00 8.00 8.65 8.65 8.65 8.65 8.55 8.75 9.35 10.05 10.00 11.15 11.35 11.90 12.00
2.45 3.00 2.10 3.35 4.00 3.35 3.30 3.95 3.95 4.00 3.95 4.00 4.65 5.75 4.00 5.35 5.35 4.65 5.35 5.40 6.00 6.00 6.10 6.50
```

```
6.00 4.90 6.00 6.65 6.75 5.65 6.65 7.35 7.35 7.35 8.00 8.00 8.65 8.65 8.65 8.65 8.65 9.20 9.35 11.05 10.65 10.00 11.30);
          array d[2,29] $30 (
' ' ' ' '|' ', .' '''/\il' '!()-:;[]`fjt' 'Is' '{}' ' ' 'cerz' ' ' ' ' '#$*0123456789?J_agovxy'
'~' ' ' '"Sbdhknpqu' ' ' '+<=>' '^' 'FP' 'BELTZ' 'ACDNRUVXYw' 'GHKOQ' '&m' ' ' ' ' '@' 'M' '%W'
''''' '|' ' ' ', .' '/:;\ijlt' '!()-I[]`fr' 'Js' ' ' '"' '?acez' '^' '{}' '#$*0123456789_bdghknopquvxy'
' ' '~' 'FPS' '+<=>' ' ' ' ' 'ELTZ' 'BCR' 'ADGHKNOQUVXYw' '&m' '%' 'M' '@' ' ' 'W' ' ');
          array v[29]   (1.79 1.99 2.19 2.49 2.76 3.32 3.88 3.93 4.07 4.42 4.68 4.79 4.99
          5.19 5.40 5.55 5.63 5.68 5.80 6.09 6.66 7.21 7.76 8.32 8.88 9.19 9.29 9.42 9.99);
          array WD[3] WD1-WD3;
    length string $1250;
    if phy^=2 then do i=1 to 47;
                    WD1 + countc(string,c[i],'T')*w[fsize-6,bold,i];
    end;
    if phy^=1 then do;
                    do i=1 to 29; WD2 + countc(string,d[bold,i],'T')*v[i];
                    end;
                    WD2 = floor(WD2*fsize*2)/20;
                    if phy=3 then WD3 = max(WD1,WD2);
    end;
    return(WD[phy]);
  endfunc;
run; quit;
options cmplib=work.funcs;

%macro Char_y_m;

%local i j;

options orientation=portrait papersize=letter topmargin=0.5in bottommargin=0.5in
          leftmargin=0.20in rightmargin=0.20in;

data Char_y_m;
          array C[4] $1 _temporary_('|' '''' 'y' 'm');
          array P[4] _temporary_(2.05 1.75 5.1 7.7);
          array L[4] _temporary_;
          array S[4] $400 Str1-Str4;
          retain Str0 '-----';
          do i=1 to 4;
                    L[i] = floor(57/P[i])*10;
                    S[i] = repeat(C[i],L[i]-1);
                    call symputx(cat('Len',i),L[i],'L');
                    call symputx(cat('Wdt',i),P[i]*L[i]+0.15,'L');
          end;
run;

%style(stylen=1, boldn=2, frame=2, rules=1, fontsize=10, margin=15
           , cellpadding=0, borderspacing=0.15, borderwidth=0.15);

title j=c "Font: Times New Roman/Normal, 10pt -- Character '|', "" ' ""(single quote), 'y' and 'm'";

ods listing close;
ods pdf file="&path\Char_y_m.pdf" style=Mystyle1_2 startpage=no dpi=1440 notoc compress=0;

proc report data=Char_y_m nowd;
    column %do i=1 %to 4; Str&i=Str&i.A Str0 Str&i=Str&i.B Str0 %end;;
          define Str0 /display style=[just=l cellwidth=19pt];
  %do i=1 %to 4; %do j=1 %to 2;
          define Str&i%scan(A B,&j) /display style=[just=l cellwidth=%sysevalf(&&Wdt&i-(&j-1)/20)pt]
                     "Str&i:&&Len&i:%sysevalf(&&Wdt&i-(&j-1)/20)pt" %if &i&j>11 %then page;;
  %end; %end;
run;

ods pdf close;
ods listing;

%mend Char_y_m;

%macro Wc_Wa;

%local i j WC;

%let WC = 500;

options orientation=portrait papersize=letter topmargin=0.5in bottommargin=0.5in
          leftmargin=0.375in rightmargin=0.375in;

data Wc_Wa;
          length char $1 string $100;
          retain char '|';
          string = repeat('0',100-1);
run;

%style(stylen=2, boldn=2, frame=1, rules=2, fontsize=10, margin=27
           , cellpadding=0, borderspacing=0, borderwidth=0);

title j=c "String as 100 of '0', coding column width from 499.97pt to 500.03pt";
footnote;

ods _all_ close;
ods pdf file="&path\Wc_Wa.pdf" style=Mystyle2_2 startpage=no dpi=1440 notoc compress=0;

proc report data=Wc_Wa nowd;
    column %do j=1 %to 7; string=str%substr(ABCDEFG,&j,1) char %end;;
          define char /display style=[just=l cellwidth=2pt] ' ';
  %do j=1 %to 7;
```

```
          define str%substr(ABCDEFG,&j,1) /display style=[just=l cellwidth=%sysevalf(&WC+(&j-4)/100)pt]
                    "&j:%sysevalf(&WC+(&j-4)/100)" %if &j>1 %then page;;
  %end;
run;

ods pdf close;
ods listing;

%Location(fname=&path\Wc_Wa);

data Wc_Wa(keep=K1 Wc Wa FL);
          set _Location_(where=(Text^=:'String as'));
          by K1 K2;
          retain ST Wc Wa FL;
          if first.K1 then do;
                    ST = Loc1;
                    Wa = . ;
                    FL = '*';
          end;
          if len=100  then FL = ' ';
          if Text='.' then Wa = Loc1 - ST;
          if last.K1;
          Wc = K2;
          Wa = Loc1 - ST;
run;

%mend Wc_Wa;

%macro Border(Test=);

%local h i j pt;

options orientation=portrait papersize=letter topmargin=0.5in bottommargin=0.5in
          leftmargin=0.375in rightmargin=0.375in;

data Border;
          array C[0:9] $100 C0-C9;
          retain char '|';
          do i=0 to 9;
                    C[i] = repeat(cat(i),100-1);
          end;
run;

%style(stylen=3, boldn=2, frame=2, rules=1, fontsize=10, margin=27
           , cellpadding=0, borderspacing=0, borderwidth=0);

ods listing close;
ods pdf file="&path\Border&Test..pdf" style=Mystyle3_2 startpage=no dpi=1440 notoc compress=0;

title j=c "Font: Times New Roman/Normal, Size: 10pt -- Border&Test..pdf";

%do i=0 %to 9; %let BD = %sysevalf(0.02*&i);

proc report data=Border nowd style=[%scan(borderspacing=&BD.pt borderwidth|cellpadding,&Test,|)=&BD.pt];
    column %do j=1 %to 2; C&i=C&i%substr(AB,&j,1) char %end;;
          define char /display style=[just=l cellwidth=5pt] ' ';
  %do j=1 %to 2;
          %let WD = %sysevalf(500+(&j-4)/100+&Test*%sysfunc(round(&BD+(0<&BD & &BD<0.025)*0.025,0.05)));
          define C&i%substr(AB,&j,1) /display style=[just=l cellwidth=&WD.pt]
                    "&i:&j:&WD:&BD" %if &j>1 %then page;;
  %end;
run;

%end;

ods pdf close;
ods listing;

%Location(fname=&path\Border&Test);

data Border&Test(keep=K1 K4 ST Wc Wa FL);
          set _Location_(where=(Text^=:'Font: Times New Roman'));
          by K1 K2;
          retain Wa Wc F1 FL ST;
          if first.K1 then do;
                    call missing(Wa,Wc,F1);
                    FL = '*';
                    ST = Loc1;
          end;
          if F1=. then do;
                    if first.K2 then K2 = 0;
                    else if last.K2 then do;
                              if FL=' ' then do;
                                        Wa = Loc1 - ST;
                                        Wc = K3;
                                        F1 = 1;
                              end;
                    end;
                    else if len=100  then FL = ' ';
          end;
          if last.K1;
run;

%mend Border;
```

```
%macro Textalign;

%local i j W;

options orientation=portrait papersize=letter
          topmargin=0.5in bottommargin=0.5in leftmargin=0.5in rightmargin=0.5in;

%style(stylen=5, boldn=2, frame=1, rules=2, fontsize=10, margin=36
           , cellpadding=0, borderspacing=0.05, borderwidth=0.05);

data Text;
          array c[3] $1 _temporary_('''' '|' '~');
          array w[3] _temporary_(1.75 2.05 5.40);
          array S[3] $2000 S1-S3;
          do i=1 to 3;
                    Num  = floor(540/w[i]/10)*10;
                    S[i] = repeat(c[i],Num-1);
                    S[i] = catt(' S={textalign=left}',S[i],' S={textalign=right} n',S[i]);
                    call symputx(cat('Num',i),Num,'L');
                    call symputx(cat('Wdt',i),Num*w[i],'L');
          end;
run;

title j=c "Font: Times New Roman/Normal, 10pt -- Left/Right Alignment";

ods listing close;
ods pdf file="&path\Textalign.pdf" style=Mystyle5_2 startpage=no dpi=1440 notoc compress=0;

proc report data=Text nowd;
    column %do i=1 %to 3; %do j=1 %to 3; S&i=S&i%scan(A B C,&j) %end; %end;;
  %do i=1 %to 3; %do j=1 %to 3; %let W = %sysevalf(&&Wdt&i+(&j=1)*10-(&j=3)*0.05);
          define S&i%scan(A B C,&j) /display style=[just=l cellwidth=&W.pt]
                    "S&i%scan(A B C,&j) Char: &&Num&i Width: &W.pt" %if &i&j>11 %then page;;
  %end; %end;
run;

ods pdf close;
ods listing;

%Location(fname=&path\Textalign);

%mend Textalign;

%macro Formula;

options orientation=landscape papersize=(22in 17in)
          topmargin=0.05in bottommargin=0.05in leftmargin=0.125in rightmargin=0.125in;

%style(stylen=4, boldn=2, frame=1, rules=2, fontsize=10, margin=9
           , cellpadding=0, borderspacing=0.05, borderwidth=0.05);

data Text;
          length String $1000;
          do i=1 to floor(1566/2.05);
                    String = repeat('|',i-1);
                    output;
          end;
run;

title "Verify Formula: W = n*p";
footnote;

ods _all_ close;
ods pdf file="&path\Formula.pdf" style=Mystyle4_2 startpage=yes dpi=1440 notoc compress=0;

proc report data=Text nowd;
    column String;
          define String /display style=[just=r cellwidth=1566pt];
run;

ods pdf close;
ods listing;

%Location(fname=&path\Formula);

data Formula;
          set _Location_(drop=K1-K5 where=(Text=:"|"));
          if abs(Loc1+len*2.05-1566)>1e-9 then
          put 'ERR' 'OR: on Formula Loc1+len*2.05=1566 ' Loc1= len=;
run;

%mend Formula;

%macro PhyscalW(start=,stop=);

%local h i D Vars;

%let D = %eval(&stop-&start+1);

options nocenter orientation=portrait papersize=(17in 22in)
          topmargin=0.5in bottommargin=0.5in leftmargin=0.125in rightmargin=0.125in;

data PDF_1;
          length String $100;
          do i=32 to 126;
```

```
                    String = ifc(i=32,'.'||repeat(byte(i),100-3)||'.',repeat(byte(i),100-1));
                    output;
          end;
run;

%do h=&start %to &stop;
%do i=1 %to 2;      %let Vars = &Vars W&h&i;

%style(stylen=&h, boldn=&i, fontsize=&h, frame=1, rules=2,
            margin=9, cellpadding=0, borderspacing=0.05, borderwidth=0.05);

title "Font: &h.pt/%scan(Bold Normal,&i)";

ods listing close;
ods pdf file="&path\PDF&h&i..pdf" style=Mystyle&h._&i startpage=no dpi=1440 notoc compress=0;

proc report data=PDF_1 nowd;
    column String;
          define String /display style=[just=r cellwidth=1206pt] "Font: &h.pt Width: 1206pt";
run;

ods pdf close;
ods listing;

%Location(fname=&path\PDF&h&i);

data W&h&i(keep=Ascii Char W&h&i Text);
          set _Location_(drop=K1-K5 where=(Text^=:"Font:"));
          length Char $1;
          if len=100 then Char = char(Text,2);
          else Char = choosec(whichc(substr(Text,1,4),'\050','\051','\134'),'(',')','\');
          Ascii = rank(Char);
          W&h&i = (1206-Loc1)/100;
          label W&h&i="%scan(Bold Norm%scan(al,1+(&D>8)),&i)";
run;

%end;
%end;

data path.Char_W;
          merge &Vars;
          by Ascii;
run;

proc sort data=path.Char_W out=Char_W;
          by &Vars;
run;

data path.Physcal_W&start(keep=Str &Vars);
          length Char $1 Str $30;
          set Char_W;
          by &Vars;
          retain Str ID;
          if first.W&stop.2 then call missing(Str,ID);
          ID + 1;
          substr(Str,ID,1) = Char;
          if last.W&stop.2;
          format &Vars 5.2;
          label Str='Characters';
run;

data Physcal_W;
          set path.Physcal_W&start end=eof;
          retain Len;
          Str = tranwrd(trim(Str),' ','(Space)');
          Len = max(Len,length(Str));
          if eof then do;
                    call symputx('Len',Len*4+0.6,'L');
                    call symputx('No',_N_,'L');
          end;
run;

%if &start=7 & &stop=12 %then %do;

data Physcal_W0;
          set path.Physcal_W&start;
          length Str2 $32;
          Grp  = _N_;
          Str2 = quote(catt(Str),"'");
          put Str2 @;
          if _N_=24 then put;
run;

proc transpose data=Physcal_W0 prefix=COL out=Physcal;
          id Grp;
          idlabel Str;
          var &Vars;
run;

data _null_;
          set Physcal;
          array C[*] COL1-COL&No;
          do i=1 to &No;
                    put C[i] @;
                    if i in(24,&No) then put;
```

```
          end;
run;

%end;

%let S = %sysevalf(7.8*72-&Len-2*&D*18.6);
%if %sysevalf(&S<0) %then %do;
          %put NOTE: The range of font size &start.pt to &stop.pt is too wider to print.;
          %return;
%end;
%let W2 = %sysfunc(round(18.5+&S/(2*&D+1),0.1));
%let W1 = %sysevalf(7.8*72-0.1-(&W2+0.1)*2*&D);

options center orientation=portrait papersize=Letter
          topmargin=0.40in bottommargin=0.40in leftmargin=0.35in rightmargin=0.35in;

%style(stylen=1, boldn=2, frame=1, rules=2, fontsize=8, margin=25.2
           , cellpadding=0.25, borderspacing=0.05, borderwidth=0.05);

title "A Character's Physical Width (Unit: pt) for Both Normal and Bold From &start.pt To &stop.pt";

ods escapechar='^';
ods listing close;
ods pdf file="&path\Physcal_W&start..pdf" style=Mystyle1_2 startpage=no dpi=1440 notoc compress=0;

proc report data=Physcal_W nowd;
    column Str %do h=&start %to &stop; ("Size: &h.pt" W&h.1 W&h.2) %end;;
          define Str   /display style=[just=l cellwidth=&W1.pt];
  %do h=&start %to &stop;
          define W&h.1-W&h.2 /display style=[just=d cellwidth=&W2.pt];
  %end;
run;

ods pdf close;
ods listing;
ods escapechar=' ';

%mend PhyscalW;

%macro V_Fontsize;

%local h i j k X;

options orientation=landscape papersize=(22in 17in)
          topmargin=0.5in bottommargin=0.5in leftmargin=0.125in rightmargin=0.125in;

%let X = 1490;

%do i=1 %to 2;

data Fontsize&i;
          array VW[21] _temporary_(1.55 1.87 2.18 2.49 3.12 3.74 4.05 4.37 4.68 4.99
                                                    5.30 5.62 5.93 6.55 7.18 7.49 8.12 8.74 9.05 9.37 9.99);
          array CW[2,21] $30 _temporary_(
          ' ' ' ' '|' ' '',./\il' '!()-:;[]`fjt' 'Is{}' ' ' 'cerz' ' ' '#$*0123456789?J_agovxy~'
          '"Sbdhknpqu' '+<=>^' 'FP' 'BELTZ' 'ACDNRUVXYw' 'GHKOQ' '&m' ' ' '@' 'M' '%W'
          '''' '|' ' ' ' ,./:;\ijlt' '!()-I[]`fr' 'Js' '"' '?acez' '^{}' '#$*0123456789_bdghknopquvxy'
          'FPS~' '+<=>' 'ELTZ' 'BCR' 'ADGHKNOQUVXYw' '&m' '%' 'M' '@' 'W' ' ');
          array S[95] $32767 S1-S95;
          length char $1;
          do h=1 to 95;
                    char = byte(h+31);
                    do i=1 to 21;
                              if CW[&i,i]>' ' & (char=' ' & i=4 | find(CW[&i,i],char,'T')>0) then leave;
                    end;
                    X  = &X*10/VW[i];
                    A  = ifn(X<2520,360,2520);
                    M  = min(5040,floor(X/A)*A);
                    FL = 0;
                    call symputx(cat("LL&i",h),cat('ASCII=',h+31,':L=',M),'L');
                    do i=1 to 12;
                              N          = M/i;
                              N2         = floor(N);
                              S[h] = catt(S[h],' S={fontsize=',i,'pt',choosec(1+((N2=N)=FL)*(1+FL),'}',' color=red}','
color=black}')
                                                  ,ifc(i=1,' ',' n'),ifc(Char=' ','.'||repeat(Char,N2-3)||'.',repeat(Char,N2-1)) );
                              FL         = (N2<N);
                    end;
                    S[h] = catt(S[h],' nEnd');
          end;
run;

data V_Fontsize&i;
          set Fontsize&i;
          label %do j=1 %to 95; S&j="&&LL&i&j" %end;;
run;

%style(stylen=5, boldn=&i, frame=1, rules=2, margin=9
           , cellpadding=0, borderspacing=0, borderwidth=0);

title j=c "Font: Times New Roman/%scan(Bold Normal,&i), Size: 1pt to 12pt -- Vision Width";

ods listing close;
ods pdf file="&path\V_Fontsize&i..pdf" style=Mystyle5_&i startpage=no dpi=1440 notoc compress=0;
```

```
proc report data=V_Fontsize&i nowd;
    column S1-S95;
          define S1     /display style=[just=l cellwidth=1566pt];
          define S2-S95 /display style=[just=l cellwidth=1566pt] page;
run;

ods pdf close;
ods listing;

%end;

%mend V_Fontsize;

%macro Vision_W;

options orientation=portrait papersize=(17in 11in)
          topmargin=0.5in bottommargin=0.5in leftmargin=0.15in rightmargin=0.15in;

data Vision_W;
          array VW[29] _temporary_(1.79 1.99 2.19 2.49 2.76 3.32 3.88 3.93 4.07 4.42 4.68 4.79 4.99
                                   5.19 5.40 5.55 5.63 5.68 5.80 6.09 6.66 7.21 7.76 8.32 8.88 9.19 9.29 9.42 9.99);
          array CW[2,29] $30 _temporary_(
' ' ' ' '|' ', .' '''/\il' '!()-:;[]`fjt' 'Is' '{}' ' ' 'cerz' ' ' ' ' '#$*0123456789?J_agovxy'
'~' ' ' '"Sbdhknpqu' ' ' '+<=>' '^' 'FP' 'BELTZ' 'ACDNRUVXYw' 'GHKOQ' '&m' ' ' ' ' '@' 'M' '%W'
''' '|' ' ' ', .' '/:;\ijlt' '!()-I[]`fr' 'Js' ' ' '"' '?acez' '^' '{}' '#$*0123456789_bdghknopquvxy'
' ' '~' 'FPS' '+<=>' ' ' ' ' 'ELTZ' 'BCR' 'ADGHKNOQUVXYw' '&m' '%' 'M' '@' ' ' 'W' ' ');
          array S[29] $32767 S1-S29;
          length Char $1;
          do h=1 to 29;
                    S[h] = ' ';
                    do i=1 to 2;
                              if CW[i,h]=' ' then continue;
                              St = (S[h]=' ');
                              if i=2 then S[h] = catt(S[h],' S={fontweight=medium color=red}');
                              do j=1 to length(CW[i,h]);
                                        Char = substr(CW[i,h],j,1);
                                        S[h] = catt(S[h],ifc(St=j=1,' ',' n')
                                                  , ifc(Char=' ','.'||repeat(Char,120-3)||'.',repeat(Char,120-1)) );
                              end;
                    end;
          end;
run;

%style(stylen=6, boldn=1, frame=1, rules=2, fontsize=10, margin=10
           , cellpadding=0, borderspacing=0, borderwidth=0);

title j=c "Times New Roman, 10pt -- Grouping Characters with Equal Vision Width";

ods listing close;
ods pdf file="&path\Vision_W.pdf" style=Mystyle6_1 startpage=no dpi=1440 notoc compress=0;

proc report data=Vision_W nowd style(header)=[fontsize=8pt];
    column S1-S29;
          define S1     /display style=[just=l cellwidth=1200pt];
          define S2-S29 /display style=[just=l cellwidth=1200pt] page;
run;

ods pdf close;
ods listing;

%mend Vision_W;

%macro V_Width;

options orientation=portrait papersize=(22in 17in)
          topmargin=0.125in bottommargin=0.125in leftmargin=0.125in rightmargin=0.125in;

data V_Width;
          array VW[29] _temporary_(1.79 1.99 2.19 2.49 2.76 3.32 3.88 3.93 4.07 4.42 4.68 4.79 4.99
                                   5.19 5.40 5.55 5.63 5.68 5.80 6.09 6.66 7.21 7.76 8.32 8.88 9.19 9.29 9.42 9.99);
          array CW[2,29] $30 _temporary_(
' ' ' ' '|' ', .' '''/\il' '!()-:;[]`fjt' 'Is' '{}' ' ' 'cerz' ' ' ' ' '#$*0123456789?J_agovxy'
'~' ' ' '"Sbdhknpqu' ' ' '+<=>' '^' 'FP' 'BELTZ' 'ACDNRUVXYw' 'GHKOQ' '&m' ' ' ' ' '@' 'M' '%W'
''' '|' ' ' ', .' '/:;\ijlt' '!()-I[]`fr' 'Js' ' ' '"' '?acez' '^' '{}' '#$*0123456789_bdghknopquvxy'
' ' '~' 'FPS' '+<=>' ' ' ' ' 'ELTZ' 'BCR' 'ADGHKNOQUVXYw' '&m' '%' 'M' '@' ' ' 'W' ' ');
          array S[29] $1000 S1-S29;
          length Char $1;
          do h=1 to 29;
                    Cn         = floor(1515/VW[h]/5)*5;
                    WD         = Cn*VW[h];
                    Bn         = 2 - (CW[2,h]=' ');
                    Char = first(CW[Bn,h]);
                    S[h] = repeat(Char,Cn-1);
                    call symputx(cat('VW',h),VW[h],'L');
                    call symputx(cat('CN',h),Cn,'L');
                    call symputx(cat('BN',h),Bn,'L');
                    call symputx(cat('WD',h),WD,'L');
          end;
run;

%style(stylen=6, boldn=2, frame=1, rules=2, fontsize=10, margin=9
           , cellpadding=0, borderspacing=0.15, borderwidth=0.15);

title j=c "Times New Roman, 10pt -- Measuring Character's Vision Width by: V_Width.pdf";
```

```
ods listing close;
ods pdf file="&path.V_Width.pdf" style=Mystyle6_2 startpage=no dpi=1440 notoc compress=0;

proc report data=V_Width nowd;
    column %do i=1 %to 29; S&i S&i=T&i %end;;
  %do i=1 %to 29; %do j=1 %to 2;
          define %scan(S T,&j)&i /display style=[just=l fontweight=%scan(bold medium,&&BN&i)
                    cellwidth=%scan(1566 &&WD&i,&j,' ')pt] "&i:&j:&&BN&i:&&CN&i%scan(:1566 *&&VW&i=&&WD&i,&j,' ')"
                    %if &i&j>11 %then page;;
  %end; %end;
run;

ods pdf close;
ods listing;

%mend V_Width;

%macro CLNMSG2;

%local i m Vn;

data CLNMSG;
    set sashelp.CLNMSG(obs=4);
run;

proc contents data=CLNMSG out=Vars(keep=VARNUM NAME LABEL) noprint;
run;

proc sql noprint;
    select distinct VARNUM, NAME, NAME as NAME2, pwidth(coalescec(LABEL,NAME),1,10,1)
      into :VN1-:VN99, :VAR separated by ' ', :VR1-:VR99, :LBL1-:LBL99
      from Vars;
%let Vn = &sqlobs;

    select %do i=1 %to &Vn; %if &i>1 %then,;
           max(&&LBL&i,max(pwidth(cat(&&VR&i),2,10,1)))+2*4 %end;
      into %do i=1 %to &Vn; %if &i>1 %then,; :WD&i trimmed %end;
      from CLNMSG;
quit;

options orientation=landscape papersize=Letter
          topmargin=0.75in bottommargin=0.75in leftmargin=0.7in rightmargin=0.7in;

%style(stylen=6, boldn=2, frame=1, rules=2, fontsize=10, margin=50
          , cellpadding=4, borderspacing=0.75, borderwidth=0.75);

ods listing close;
ods pdf file="&path\CLNMSG.pdf" style=Mystyle6_2 startpage=no dpi=1440 notoc compress=0;

title j=c "(13) Set Column Width by function pwidth()";

%do m=0 %to 1;

proc report data=CLNMSG nowd;
    column &VAR;
  %do i=1 %to &Vn;
    define &&VR&i /display style=[just=%scan(l c,1+%sysevalf(&&WD&i<57))
                                  cellwidth=%sysevalf(&&WD&i-&m*0.05)pt];
  %end;
run;

%end;

ods pdf close;
ods listing;

%mend CLNMSG2;

%Char_y_m;
%Wc_Wa;
%Border(Test=1);
%Border(Test=2);
%Textalign;
%Formula;
%PhyscalW(start=7,stop=12);
%V_Fontsize;
%Vision_W;
%V_Width;
%CLNMSG2;

proc fcmp outlib=work.funcs.maths;
    deletefunc pwidth;
run;
options cmplib=();
```