Get the following data sets from the course web site

- `patient.txt`, space separated data items, no header
- `patient.csv`, single sheet csv spread sheet, no header
- `patient_names.csv`, single sheet csv spread sheet, with header line giving variable names
- `ice.txt`, space separated data items, with header

Save them to a `data` folder that you create on the UDrive `U:\data` on the terminal server.

Other data file formats, e.g., fixed column format, are possible, but we won’t deal with them here. Consult *Learning SAS by Example* by Ron Cody on other data formats.
We need to take a few steps to format our data before reading them via the data step.

- Use a text editor (Notepad) to make any (global) changes on delimiters and missing values.
- The SAS default delimiter is a blank " ", or several blanks between individual data items.
- Header Rows: Data sets should not have header rows. If you have a header row, you can skip it by using `infile "U:\data\patient.txt" firstobs=2;`
- We specify the header names as well as the data type explicitly in the data step.
- Missing Values: We must find any missing values or NA’s and convert them to a period “.” for SAS to recognize as such.
- The period must be be separated from other values by one or more spaces.
- Separate adjacent missing values by spaces as well.
Reading Data Values Separated by Spaces

- Read and print the data in `patient.txt` to the screen.

```sas
data patient1; * data set name;
   infile "U:\data\patient.txt";
   input ID Age Sex $;
   run;

title "Patient DATA 1";
proc print data= patient1;
run;
```

- See what happens when you replace two adjacent values in `patient.txt` by two periods without a space in between.
- When you have I/O questions, experiment with the feature in question on some small data set.
- Use the `dsd` (Delimiter-Sensitive Data) option in `infile`.
- Read and print the data in `patient.csv` to the screen.

```sas
data patient2; * data set name;
  infile "U:\data\patient.csv" dsd;
  input ID Age Sex $;
  run;
title "Patient DATA 2";
proc print data= patient2; run;
```

- Changes default delimiter to a comma.
- Assumes missing values for empty slots.
  No need for periods to indicate missing values.
- Character values in quotes have the quotes stripped off.
- For a file `fname.txt` with other delimiters like `:` use
  `infile "U:\data\fname.txt" dsd dlm= '"';` instead.
The Need for Permanent SAS Data Sets

- SAS procs only work on SAS data sets, which are created with the data input step.
- They are temporarily stored in the WORK library folder.
- After a SAS session closes these data sets are gone. They need to be recreated for each new SAS session.
- This would require another data input step.
- No big deal for small data sets, but for large ones it would be preferable to have a SAS data set from the start.
libname mydata "U:\data"; * an existing location;
data mydata.patient4;
infile "U:\data\patient.csv" dsd;
input ID Age Sex $;
run;
title "Patient Data 4";
proc print data=mydata.patient4;
run;

- These lines create the permanent SAS data set `patient4 U:\data\patient4.sas7bdat`.
- That data set also appears in the temporary Library folder `Mydata`. `Mydata` disappears after the end of a SAS session, but `U:\data\patient4.sas7bdat` is still there.
- `Mydata` acts as a shortcut handle for `U:\data`.
- Instead of the libref `mydata` you can use any other proper SAS name with ≤ 8 characters.
When you delete `U:\data\patient4.sas7bdat` it also disappears from the temporary Library folder `Mydata`.

When you delete `patient4` from the temporary Library folder `Mydata` it also disappears from `U:\data`.

If you rename it to `U:\data\patient5.sas7bdat`, it also renames to `patient5` in `Mydata`, after a refresh, by stepping out and back into the `Mydata` library.

In a later SAS session or in the same session you can access `patient4` by giving another libref statement, e.g.,

```
libname mydata2 "U:\data";
```

and use `mydata2.patient4` wherever you used `mydata.patient4` before.

View `mydata` or `mydata2` as conduits to `U:\data`, and whatever you do (delete or rename) w.r.t. any SAS data set in one it is also done in the other. Play around with this.
Prior to using a permanent data set, such as patient4, in a new SAS session, you need an appropriate libname statement, i.e., you need a conduit, e.g., in a new SAS session try

```sas
libname mydata "U:\data";
title "Patient Data 4";
proc print data=mydata.patient4;
run;
```

SAS needs to know where to find a permanent SAS data set.

Running simply the first line above, you can look at the data via SAS Explorer ⇒ Libraries ⇒ the newly created folder Mydata ⇒ double click patient4, which opens up VIEWTABLE on that file.
The following code saves the permanent SAS data set patient4.sas7bdat in folder U:\data to a file U:\data\odsexample.csv

libname mydata 'U:\data';
ods csv file='U:\data\odsexample.csv';
proc print data=mydata.patient4 noobs; run;
ods csv close;

- ODS stands for Output Delivery System
- The ODS CSV opens the CSV file as an output destination.
- PROC PRINT \(\rightarrow\) patient4 data \(\rightarrow\) output destination.
- the NOOBS in PROC PRINT removes the observation column.
- !!! Close file with ODS CLOSE following PROC PRINT.
Using some Common Procs

libname mydata "U:\data";
title "Gender Frequencies";
proc freq data=mydata.patient4;
  table Sex; run;

Without the variable after table you get

ERROR 22-322: Syntax error, expecting one of the following: a name, _ALL_, _CHARACTER_, _CHAR_, _NUMERIC_

libname mydata "U:\data";
title "Age Summary";
proc means data=mydata.patient4
  n mean std median clm alpha=.005;
  var Age; run;

- Without var Age; get stats on all numeric variables.
Gender Frequencies

The FREQ Procedure

<table>
<thead>
<tr>
<th>Sex</th>
<th>Frequency</th>
<th>Percent</th>
<th>Cumulative Frequency</th>
<th>Cumulative Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>5</td>
<td>50.00</td>
<td>5</td>
<td>50.00</td>
</tr>
<tr>
<td>M</td>
<td>5</td>
<td>50.00</td>
<td>10</td>
<td>100.00</td>
</tr>
</tbody>
</table>

Age Summary

The MEANS Procedure

<table>
<thead>
<tr>
<th>Analysis Variable : Age</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
</tr>
<tr>
<td>---</td>
</tr>
<tr>
<td>10</td>
</tr>
</tbody>
</table>
libname mydata "U:\data";
title "Sorting by Sex";
proc sort data=mydata.patient4;
  by Sex; run;

- This sorts the SAS data set by Sex (also in its permanent location). Needed if you split analyses using by.
- See what happens when using by Sex Age and by Age Sex.

title "Summaries by Sex";
proc means data=mydata.patient4;
  var Age;
  by Sex; run;
* first sort by Sex alone again,
  if you tried the above: by Age Sex;
### Summaries by Sex

#### The MEANS Procedure

**Sex=F**

<table>
<thead>
<tr>
<th>N</th>
<th>Mean</th>
<th>Std Dev</th>
<th>Minimum</th>
<th>Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>20.0000000</td>
<td>4.2426407</td>
<td>14.0000000</td>
<td>24.0000000</td>
</tr>
</tbody>
</table>

**Sex=M**

<table>
<thead>
<tr>
<th>N</th>
<th>Mean</th>
<th>Std Dev</th>
<th>Minimum</th>
<th>Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>14.0000000</td>
<td>5.0497525</td>
<td>8.0000000</td>
<td>21.0000000</td>
</tr>
</tbody>
</table>
data ice;
    infile "U:\data\ice.txt" firstobs=2;
    input Heat Method $; run;
    title "Latent Heat of Fusion of Ice";
    proc print data=ice; run;
    title "Latent Heat of Fusion of Ice, Testing H: mean=80 for Method A";
    proc ttest data=ice H0=80;
        var Heat;
        where Method = "A"; run;
    title "Latent Heat of Fusion of Ice, Testing Equality of Methods A & B";
    proc ttest data = ice;
        class Method; * sorted by method first!;
        var heat; run;
Latent Heat of Fusion of Ice: Data, t-Test $H : \mu_A = 80$

<table>
<thead>
<tr>
<th>Obs</th>
<th>Heat</th>
<th>Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>79.982</td>
<td>A</td>
</tr>
<tr>
<td>2</td>
<td>80.041</td>
<td>A</td>
</tr>
<tr>
<td>3</td>
<td>80.018</td>
<td>A</td>
</tr>
<tr>
<td>4</td>
<td>80.041</td>
<td>A</td>
</tr>
<tr>
<td>5</td>
<td>80.030</td>
<td>A</td>
</tr>
<tr>
<td>6</td>
<td>80.029</td>
<td>A</td>
</tr>
<tr>
<td>7</td>
<td>80.038</td>
<td>A</td>
</tr>
<tr>
<td>8</td>
<td>79.968</td>
<td>A</td>
</tr>
<tr>
<td>9</td>
<td>80.049</td>
<td>A</td>
</tr>
<tr>
<td>10</td>
<td>80.029</td>
<td>A</td>
</tr>
<tr>
<td>11</td>
<td>80.019</td>
<td>A</td>
</tr>
<tr>
<td>12</td>
<td>80.002</td>
<td>A</td>
</tr>
<tr>
<td>13</td>
<td>80.022</td>
<td>A</td>
</tr>
<tr>
<td>14</td>
<td>80.020</td>
<td>B</td>
</tr>
<tr>
<td>15</td>
<td>79.939</td>
<td>B</td>
</tr>
<tr>
<td>16</td>
<td>79.980</td>
<td>B</td>
</tr>
<tr>
<td>17</td>
<td>79.971</td>
<td>B</td>
</tr>
<tr>
<td>18</td>
<td>79.970</td>
<td>B</td>
</tr>
<tr>
<td>19</td>
<td>80.029</td>
<td>B</td>
</tr>
<tr>
<td>20</td>
<td>79.952</td>
<td>B</td>
</tr>
<tr>
<td>21</td>
<td>79.968</td>
<td>B</td>
</tr>
</tbody>
</table>

The TTEST Procedure

Variable: Heat

<table>
<thead>
<tr>
<th></th>
<th>N</th>
<th>Mean</th>
<th>Std Dev</th>
<th>Std Err</th>
<th>Minimum</th>
<th>Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td>13</td>
<td>13</td>
<td>80.0206</td>
<td>0.0238</td>
<td>0.00660</td>
<td>79.9680</td>
<td>80.0490</td>
</tr>
</tbody>
</table>

Mean 95% CL Mean Std Dev 95% CL Std Dev
80.0206 80.0062 80.0350 0.0238 0.0171 0.0393

DF  t Value  Pr > |t|
12  3.13  0.0088
Method "A" QQ-Plot & t-Test for $H : \mu_A = \mu_B$

\[ F^* = \text{Folded F} = \max(s_1^2, s_2^2)/\min(s_1^2, s_2^2), \text{ see } \text{http://www.ats.ucla.edu/stat/sas/output/ttest.htm} \text{ near bottom.} \]

I don’t agree with SAS’s p-value calculation. It should be

\[ p\text{-value} = P(s_1^2/s_2^2 \geq F^*_\text{obs}) + P(s_1^2/s_2^2 \leq 1/F^*_\text{obs}) = 0.4423 \]

for $F^*_\text{obs} = 1.707524$. 

\[ F^* = \text{Folded F} \]
2 Sample t-test for $H_0 : "A" = "B"$
In R you would use `t.test`.

In this case SAS presents a whole bunch of pages as results, some in tabular form, some in the form of graphics.

This is typical for packages like SAS. It is a package deal!

The previous output illustrations were done by printing specific page pairs to PDF and including them via trim and clip parameters using `includegraphics` in \LaTeX.

For graphics output you can right click on the graphic and save it as a `.png` file, which you then include like any other graphic in your \LaTeX file, using `includegraphics`.

Right clicking tabular output allows saving as Excel file.

The next 3 slides show previous graphics via `.png` versions.
Latent Heat for Fusion of Ice: Data, t-Test $H : \mu_A = 80$
2 Sample t-test for $H_0 : "A" = "B"$
Methods "A" and "B" QQ-Plots

Q-Q Plots of Heat

Heat vs. Quantile for Method A
Heat vs. Quantile for Method B
ods PDF newfile=output file='U:\data\Ice1.pdf';
data ice; infile "U:\data\ice.txt" firstobs=2;
input Heat Method $; run;
title "Latent Heat of Fusion of Ice";
proc print data = ice; run;
title "Latent Heat of Fusion of Ice, Testing H: mean=80 for Method A";
proc ttest data=ice H0=80; var Heat;
where Method="A"; run;
title "Latent Heat of Fusion of Ice, Testing Equality of methods A & B";
proc ttest data=ice;
class Method; *sorted by method first!;
var Heat; run; ods PDF close;
• Putting NEWFILE=OUTPUT and a 1 in Ice1.pdf increments that number for each new output page.
• The previous code produces Ice1.pdf,..., Ice10.pdf in U:\data.
• My understanding is that each file should contain one page.
• For some reason some contain more than 1 page (???).
There are a large number of SAS Procs.

We have seen examples usages of FREQ, MEANS, SORT, and TTEST. Others of interest are: ANOVA, BOXPLOT, CORR, NPAR1WAY, PLOT, REG.

Each such Proc has quite a few usage options.

To access documentation with examples on these Procs click on SAS Procs under the next bullet.

SAS Procs or search for SAS Procs in Google, or go to