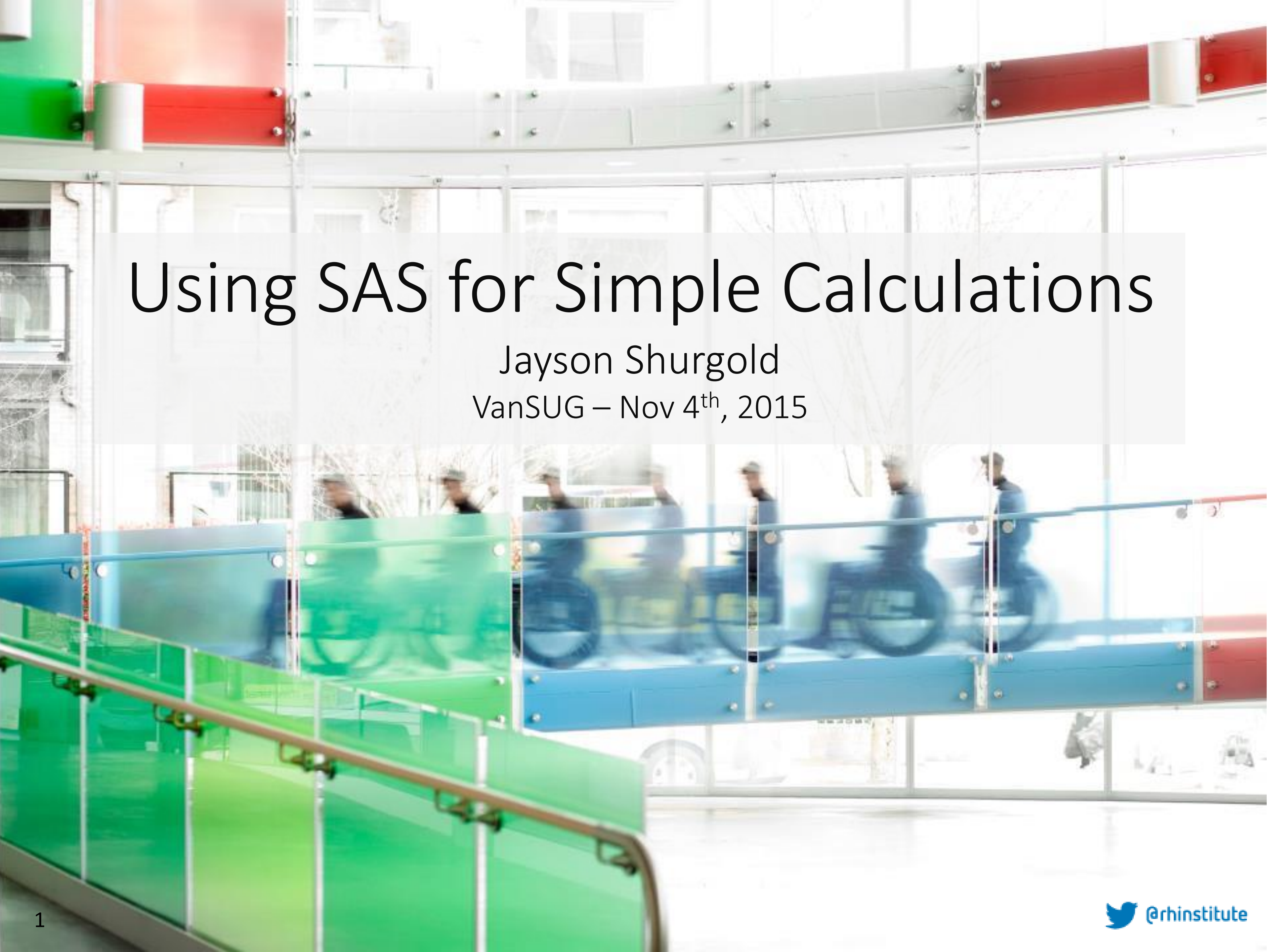


Using SAS for Simple Calculations

Jayson Shurgold

VanSUG – Nov 4th, 2015

SAS – Modern, Reliable, Accurate:

support.sas.com…

- Access data in almost any format (SAS tables, Excel, and others).
- Manage and manipulate your data (data subsets, data combinations, or new data columns).
- Data analysis using statistical techniques (descriptive measures, correlations, logistic regression, modern model selection, or Bayesian hierarchical models).
- Present results and generate reports.

- Calculations!

Comparison: Calculator vs PC (SAS)

Image to scale

Comparing simple calculations

Example 1: $X = 50 + 6$

Regular Calculator

PC with SAS

Example 1: X = 50 + 6

Regular Calculator

PC with SAS

```
data calculation;
  x=50+6;
  put 'The answer is:' x;
  run;
```

```
68    data calculation;
69    x=50+6;
70    put 'The answer is:' x;
71    run;

The answer is:56
```

Comparing simple calculations

Example 2: X = (9/45)(5*8)+9(42)

Regular Calculator

PC with SAS

Example 2: X = (9/45)(5*8)+9(42)

Regular Calculator

PC with SAS

```
data calculation;
x=(9/45)*(5*8)+9*(42);
put 'The answer is:' x;
run;


80     data calculation;
81     x=(9/45)*(5*8)+9*(42);
82     put 'The answer is:' x;
83     run;

The answer is:386
```

Example 3: X = (3x0.1) – 0.3

Regular Calculator

PC with SAS

Comparing simple calculations

Example 3: X = (3x0.1) – 0.3

Regular Calculator

PC with SAS

```
data calculation;
x=(3*0.1)-0.3;
put 'The answer is:' x;
run;


96      data calculation;
97      x=(3*0.1)-0.3;
98      put 'The answer is:' x;
99      run;

The answer is:5.551115E-17
```

Everyday Numerical Errors:

Example 1: Base 10

Fraction	Decimal (base 10)
2/1	2
1/1	1
1/2	0.5
1/3	0.33333
1/4	0.25

Example 2: ???

x	y
1.57	1.55
60.0	60.0
2.35	2.35
8.88	8.90
1.00	1.00

A global programming problem:

Most programming languages do not understand recursion:

$$\frac{1}{3}+\frac{1}{3}+\frac{1}{3}=1$$

$$\frac{1}{3}+\frac{1}{3}+\frac{1}{3}\neq 1$$

Numerical precision in general

Recall: (3x0.1) – 0.3

Using decimal arithmetic, the value 0.1 has an exact representation
Using binary arithmetic, the value 0.1 does not have an exact representation

0.0001 1001 1001 1001 100 1 1001 1001 1001 1001 100

At some point, the value represented is truncated or rounded, leading to error

@rhinstitute

How does SAS store numbers:

Consider the number 987, which can be also be expressed as: 0.987×10^3

$$\underbrace{(+)}_{\text{Sign}}\;\underbrace{0.987}_{\text{Mantissa}} \times \underbrace{10}_{\text{Base}}\;\underbrace{{}^{3}}_{\text{Exponent}}$$

Sign: Positive or Negative
Mantissa: Represents the number to be multiplied by the Base
Base: The number being raised to a power
Exponent: The power to which the base is raised

How does SAS store numbers:

Consider the number 987, which can be also be expressed as: 0.987×10^3

$$\underbrace{(+)}_{\text{Sign}}\ \underbrace{0.987}_{\text{Mantissa}} \times \underbrace{10}_{\text{Base}}\,{}^{\underbrace{3}_{\text{Exponent}}}$$

Another Day

8 or 9 step process

Bias of 127, implicit digits

$$\underbrace{0}_{\text{Sign}}\ \underbrace{100\;0100}_{\text{Exponent}}\ \underbrace{0111\;0110}_{\text{Mantissa}}$$

Floating Point Representation

How does SAS store numbers:

Remember that the default length of a numerical variable in SAS is 8 bytes

	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8
	8 bits	8 bits	8 bits	8 bits	8 bits	8 bits	8 bits	8 bits
987 =	01000100	01110110	11000000	00000000	00000000	00000000	00000000	00000000
0.1 =	00111101	11001100	11001100	11001100	11001100	11001100	11001100	11001101

Exp

Sign

Mantissa

Error

Why Floating Point Representation?

Space and Time.

Floating point representation allows the efficient calculation of very large and/or very small numbers using the same predictable 8 bytes.

In essence, Floating Point Representation is Scientific Notation in Base 2

How this relates to you - Decimals.

Example 1: Iterations.

```
data _null_;
do i=-1 to 1 by .1;
 put i=;
if i=0 then put 'AT ZERO';
end;
run;
```

```
i=-1
i=-0.9
i=-0.8
i=-0.7
i=-0.6
i=-0.5
i=-0.4
i=-0.3
i=-0.2
i=-0.1
i=-1.38778E-16
i=0.1
i=0.2
i=0.3
i=0.4
i=0.5
i=0.6
i=0.7
i=0.8
i=0.9
i=1
```

Example 2: Manipulations

Example 2: Data Manipulations:

```
data a;
x=15.7;
y=-11.9;
z=x+y;
if z=3.8 then put 'eligible';
else put 'not eligible';
run;
```

```
data a;
        x=15.7;
        y=-11.9;
        z=x+y;
       if z=3.8 then put 'eligible';
          else put 'not eligible';
run;

not eligible
```

One well-rounded solution:

```
data _null_;
do i=-1 to 1 by .1;
I = round(I, .1);
put i=;
if i=0 then put 'AT ZERO';
end;
run;
```

```
i=-0.7
i=-0.6
i=-0.5
i=-0.4
i=-0.3
i=-0.2
i=-0.1
i=0
AT ZERO
```

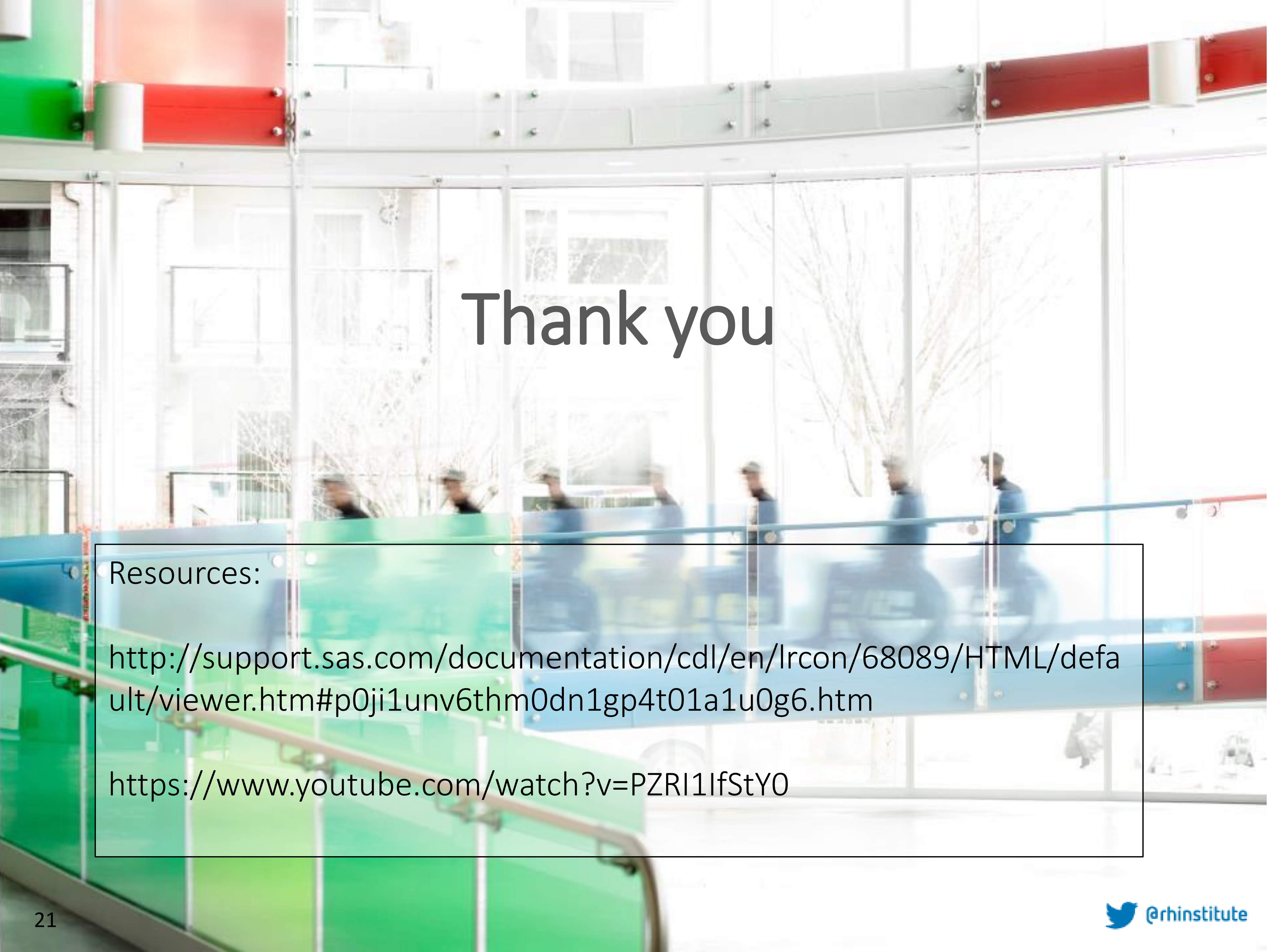
```
data a;
x=15.7;
y=-11.9;
z=round(x+y,0.1);
if z=3.8 then put 'eligible';
else put 'not eligible';
run;
```

```
119   data a;
120          x=15.7;
121          y=-11.9;
122          z=round(x+y,0.1);
123         if z=3.8 then put 'eligible';
124              else put 'not eligible';
125   run;

eligible
```

And finally… X = (3x0.1) – 0.3

```
data calculation;
x=round((3*0.1)-0.3,0.1);
put 'The answer is: ' x;
run;
```

```
data calculation;
x=round((3*0.1)-0.3,0.1);
put 'The answer is: ' x;
run;

The answer is: 0
```



Thank you

Resources:

http://support.sas.com/documentation/cdl/en/lrcon/68089/HTML/default/viewer.htm#p0ji1unv6thm0dn1gp4t01a1u0g6.htm

https://www.youtube.com/watch?v=PZRI1IfStY0