

# Saskatchewan SAS User Group

28APR2010

## SAS/IML

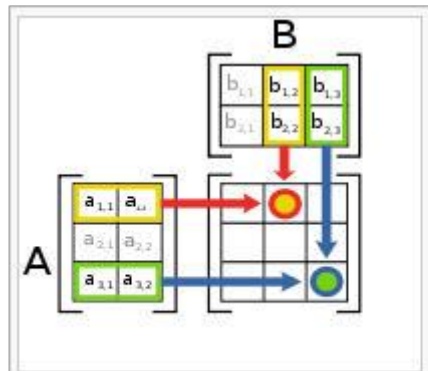
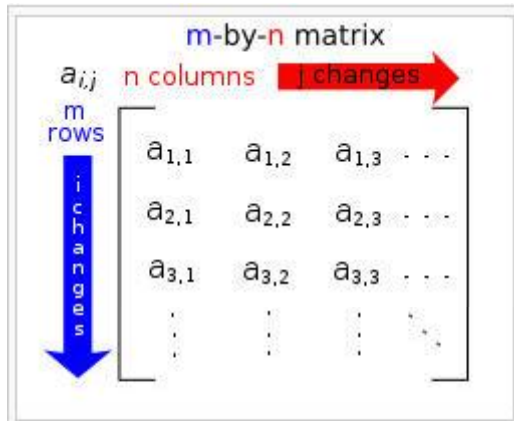
---

Sylvain Tremblay

SAS Canada – Education

**THE  
POWER  
TO KNOW<sup>®</sup>**

# SAS/IML – Interactive Matrix Language



$$[AB]_{i,j} = A_{i,1}B_{1,j} + A_{i,2}B_{2,j} + \dots + A_{i,n}B_{n,j} = \sum_{r=1}^n A_{i,r}B_{r,j}$$

In IML, the code is:  $A*B$

# Agenda

- Summary of IML
- Why would you use IML?
- Basic Syntax
- Example
- SAS/IML Studio
- Conclusion / Questions

# Summary of SAS/IML

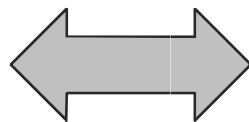
- Is a programming language
- Is Interactive
- Is Dynamic
- Is using **data matrices** as a fundamental object
- Has a powerful vocabulary of operators
- Can also produces graphics

# Why would you use IML?

Typically, because you need to:

- solve mathematical problems
- implement new statistical techniques

$$B = (X'X)^{-1}X'Y$$



```
b=inv(x`*x)*x`*y;
```

In IML code

# Why would you use IML?

Because you will save time!

- programs that take hundreds of lines of code in other languages often take only a few lines in IML!
- IML matrix functions and operators will execute more quickly than their equivalents in SAS/Base

# Basic Syntax

SAS/STAT has 72 Procedures

The **SAS/IML** module has only 1 Proc: **Proc IML**

```
Proc IML <options>;
```

```
    IML statements;
```

```
Quit;
```

Commonly used option: **WORKSIZE=**

**IML does everything in Memory !!!**

# Reading Data

- Create a Matrix from a SAS Dataset
- Create a SAS Dataset from a Matrix

## **Matrices can be:**

- Character
- Numeric

# Statements in PROC IML

## Four main categories

- Control Statements
- Commands
- Call Statements and Subroutines
- Assignment Statements

# Statements in PROC IML

## Control Statements

- IF-THEN/ELSE
- DO/END
- Iterative DO
- START/FINISH (User defined IML subroutines)

# Statements in PROC IML

## Commands

- **FREE** (free memory: remove values from matrix)
- **PRINT** (the content of a matrix)
- **READ** (data from a SAS Dataset into a matrix)
- **USE** (opens a SAS Dataset for READ access)

# Statements in PROC IML

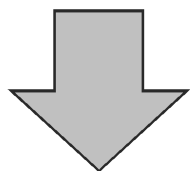
## Call Statement and Subroutines

- `CALL subroutine-name(argument1, argument2,...)`
- You can call:
  - An IML subroutine
  - A User defined subroutine

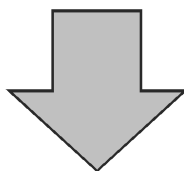
# Statements in PROC IML

## Assignment Statement

- Result = Expression



a Matrix



Operands  
Operators  
Constants  
Functions

# IML Demo - Simple Linear Regression

- $Y = B_0 + B_1X + e$

- $Y = \begin{bmatrix} 1 & X_1 \\ 1 & X_2 \\ \vdots & \vdots \\ 1 & X_n \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix} + \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix}$  in matrix notation

- B is estimated by the Least Squares

$$B = (X'X)^{-1}X'Y$$



# SAS/IML Studio



- Runs on a PC on Windows and connects with one or more SAS servers
- Available in SAS 9.2, requires BASE, STAT and IML
- **Point-and-click tool for exploratory data analysis**  
(replaces SAS/Insight)
- **Programming environment**
- Uses the IML Plus language (IML + call SAS + call R)

# In conclusion

- SAS IML is a powerful language if you need to solve mathematical problems
- More efficient than the Data Step if you are working with matrices
- Very rich function and subroutine set + the capacity to create your own
- IML Studio in which you can leverage the IML Plus language

# To learn more

## **SAS/IML Documentation**

<http://support.sas.com/documentation/cdl/en/imlug/59656/HTML/default/imlstart.htm>

## **SAS/IML Studio**

<http://support.sas.com/documentation/cdl/en/imlsstat/62555/HTML/default/statintro.htm>

## **SAS Training**

Introduction to Programming with SAS/IML Software

# Questions?



THE  
POWER  
TO KNOW®

Thanks!

Sylvain Tremblay  
[sylvain.tremblay@sas.com](mailto:sylvain.tremblay@sas.com)