



Free Platform For Numerical Computation

August, 25th, 2011

Presentation

Sylvestre Ledru

- In charge of the R&D projects
- Responsible of GNU/Linux & Mac OS X
- Developer
- Community manager for Scilab
- ... and also for IRILL
- Debian Developer

Disclaimer

- Engineer against researcher
- IT people against non-it people
- Academic against non-academic

Scilab as a consortium



History of Scilab

INSTITUT NATIONAL
DE RECHERCHE
EN INFORMATIQUE
ET EN AUTOMATIQUE



- Started in the mid 80
- Inspired by the Matlab fortran
- Fortran was too complex to handle matrices
- Needed to do some researchs at the INRIA

History of Scilab

- Developed by a research project at the INRIA since 1990
- From 2003 to 2008, through the Scilab consortium
- Since 2008, the Scilab consortium is hosted by the Digiteo foundation
- 2011 : *Scilab entreprises* created for the *classical* open source business model (most of the current employees being founders)
- Currently ~15 persons

The Consortium

Higher Education Schools



Public organizations



Companies



The Scilab Software



Scilab

- Numerical computing software
- Interpreted language
- Weakly dynamically typed
- About 2300 functions available from the language

Scilab

- Opensource (Scilab licence) since 1994 and free since Scilab 5.0 (under the CeCILL license – GPL compatible)
- Multiplatform (GNU/Linux, Mac OS X, Windows, Unix...)
- Current version: 5.3.3

Scilab

- Many libraries are binded/wrappers in Scilab
 - Hide complexity
 - Provides a common language
 - Allow interactions between incompatible libraires
 - Remove the need to know C, C++ or Fortran programmation
 - ...

Who is using Scilab (or Octave) ?

- Students in engineering
- Engineers (spatial, avionics, car industry, etc)
- Traders and bankers
- Researchers
- ...

What for ?

- Scilab can be used:
 - To develop complex applications
 - As a prototyping application
 - Link and use a load level library into a high level language
 - A powerfull calculator
 - Computing engine
 - Control external devices

Scilab - CLI

```
Paramétrage de scilab-full-bin (5.3.3-2) ...
Paramétrage de scilab (5.3.3-2) ...
Paramétrage de scilab-doc (5.3.3-2) ...
Paramétrage de scilab-sivp (0.5.3-2) ...
Traitement des actions différées (« triggers ») pour « menu »...
[23:59:08][sylvestre@losinj] ~$ scilab-cli

-----
                    scilab-5.3.3
-----

      Consortium Scilab (DIGITEO)
      Copyright (c) 1989-2011 (INRIA)
      Copyright (c) 1989-2007 (ENPC)
-----

Initialisation :
  Chargement de l'environnement de travail

-->a=2*[2,3]
a =

    4.    6.

-->
```

Scilab GUI

The screenshot displays the Scilab GUI interface. On the left is a File Browser showing the directory structure of the Scilab installation. The central Console Scilab window shows numerical data and command execution. On the right, the Historique des Commandes window shows a list of executed commands and their timestamps. Below the console is the Explorateur de Variables window, which displays a table of current variables.

Name	Dimen...	Type	Visibili...
date_st	N/A	Mlist	local
aze	1x1	Chaîne	local
a	10x10	Double	local

Scilab - Graphics + doc

Fichier Edition Préférences Contrôle Applications ?

Navigateur d'aide

Figure n°0

int8(37) → Set bit → Convert to → -91.0

Type de donnée : 5
Indice du bit : 7

Exemple

Ci-dessous un cas simple d'utilisation du bloc. [Ouvrir cet exemple dans Xcos](#)

Fonction d'interfaçage

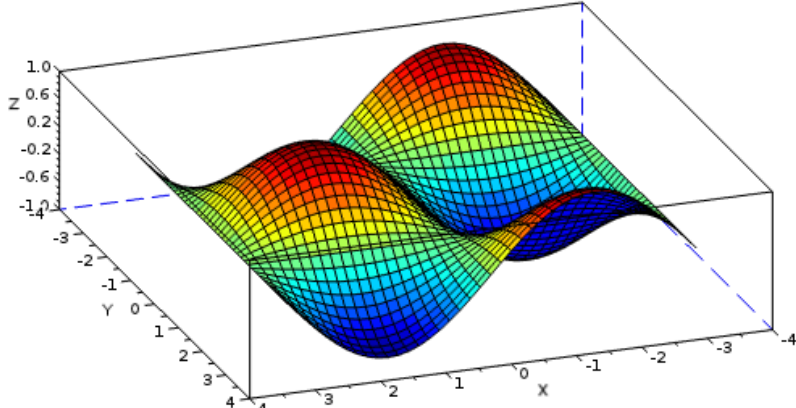
- [SCI/modules/scicos_blocks/macros/IntegerOp/BITSET.sci](#)

Fonctions de calcul

- [SCI/modules/scicos_blocks/src/c/bit_set_32.c](#)
- [SCI/modules/scicos_blocks/src/c/bit_set_16.c](#)
- [SCI/modules/scicos_blocks/src/c/bit_set_8.c](#)

Voir aussi

- [BITCLEAR](#) — Positionne un bit à 0
- [EXTRACTBITS](#) — Extraction de bits
- [LOGICAL_OP](#) — Opération logique

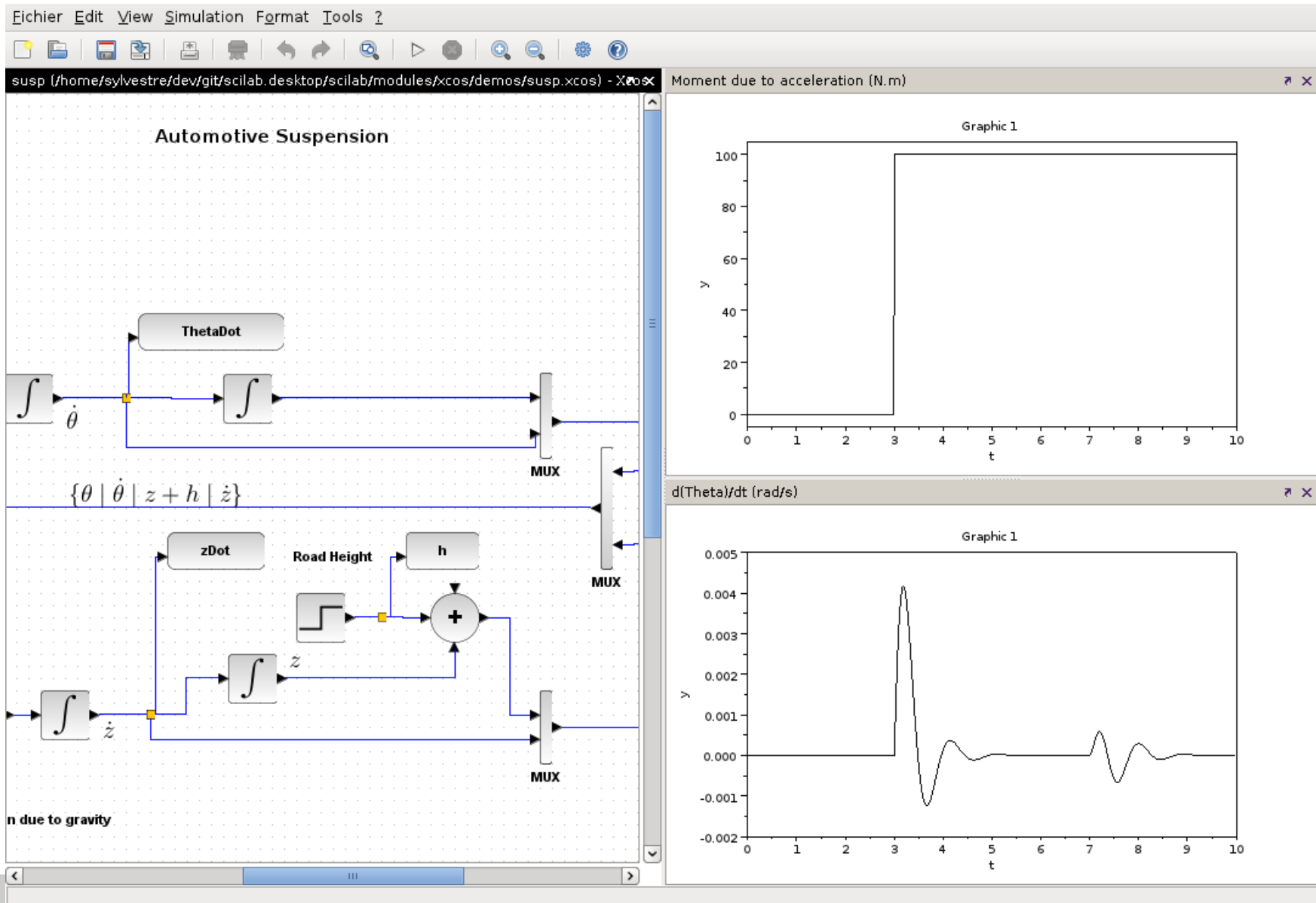


```
viewport = [0,0]
figure_name = "Figure n°%d"
figure_id = 0
info_message = ""
color_map= matrix 32x3
pixmap = "off"
pixel_drawing_mode = "copy"
anti_aliasing = "8x"
immediate_drawing = "on"
background = -2
visible = "on"
rotation_style = "unary"
event_handler = ""
event_handler_enable = "off"
user_data = []
tag = ""

-->
```

Explorateur de Variables Historique des Commandes Console Scilab

Scilab - Xcos



Scilab & Octave

Scilab vs Octave - Features

- A lot of in common
- Scilab provides an equivalent to Simulink called Xcos. A simulation and modeling for complex systems. Only free alternative in the FOSS world
- Scilab provides out of the box graphics

Scilab vs Octave - Matlab compatibility

- Octave focus on Matlab compatibility
- Scilab: Matlab is a source of inspiration when they are doing good things
- Scilab has some important differences:
 - `//` for comments instead of `%`
 - `2./ <> 2 ./`
 - Different function profiles
 - Different graphics features

Scilab vs Octave - Community

- Octave has a bigger ecosystem (toolboxes)
- ... probably because Scilab was not free for a while
- Octave has no structure behind while Scilab has full time (paid) engineers
ie : the classical « community driven » vs « integrated team driven »

Scilab for non-geeks

Extensibility objectives

- Allows users to increase the number of features
- Provide easy access to extension mechanisms

You never know what a user is going to do with your software

KISS

- Allows developers to create simple modules/toolboxes
 - From basic macros for a single function
 - To C, C++, Java or Python based modules with full documentation, unitary tests, non reg tests...
 - Provides module/toolbox skeleton

KISS - A module with only Scilab macros

- Very easy to write. Just write a .sci file in the macros/ directory
- Various mechanism to publish the code:
 - File exchange: <http://fileexchange.scilab.org/>
 - Scilab packaging system: <http://atoms.scilab.org/>
 - Forge: <http://forge.scilab.org/>

Common API is provided

- API_Scilab is a full API to manage reading/writing data from/to Scilab memory.
 - Easy to use
 - Lot of error managements (unlike Matlab with the mex)
 - Fully documented with examples
 - Unitary tests
- => Help non-experienced C or C++ developers

A module with native code

- Multiplatform code (should build and run on the three official OS)
- Provides helper functions to hide the build process with the *ilib_** functions
 - Detects the compilers (C, C++ or Fortran) on each OS
 - Launch the compilation
 - Generate some *loaders*
 - Load the new libraires

Example

```
f1=['int ext1c(int *n, double *a, double *b, double *c)'  
   '{int k;'  
   '  for (k = 0; k < *n; ++k) '  
   '      c[k] = a[k] + b[k];'  
   '  return(0);}'];
```

```
mputl(f1, 'fun1.c')
```

```
ilib_for_link('ext1c', 'fun1.c', [], "c")
```

```
exec loader.sce
```

Native module : How to handle such things

- **GNU/Linux, Mac OS X & Unix :**
Based on the autotools
Detects many compilers + options on many OS/distro
Private message : Many thanks to *Ralf Wildenhues*
- **Microsoft Windows :**
Auto-generated Visual projects

Perception for user

- We tackled the compilation issue
- Not that hard to debug

However :

- Some parts look like magic (can be frustrating for developers)
- If it is not packaged in ATOMS, it is hard for normal user to build it

Science oriented language

Nightmare for language specialist

- Global and local variables are managed in a lazy (and sometime, weird) way
- A lot of ways to do the same thing
- No scalar values : everything is matrix
- ...

How to migrate a software from the academic world to the software world?

Transition from a research project to a software editor

- From *politic* perspective
 - Objectives ?
 - New features ?
 - Roadmap
 - Time constraints

Transition from a research project to a software editor

- From the human perspective
 - Hard to change the mentalities
 - Most of the developers hate constraints!
 - Being a developer is an actual job as researcher is
 - Engineers stay longer (INRIA: 2 to 5 years)
 - Some contributors do not accept that
 - Some users do not accept that

Transition from a research project to a software editor

- From a technical perspective
 - Things are not done the same way
 - Uniformisation
 - Importance of the technological choices
 - Importance of the dependencies (libraries)
 - Clean process

Transition from a research project to a software editor

- Classic example: Inclusion of thirdparty sources into the source tree

Pro:

- Can be patched
- Do not need thirdparty libraries installed on the system (do not need of a complex ./configure)
- Do not need to interact with upstream

Con:

- Unmaintainable on a long run
- Hard to follow new upstream releases
- Some bugs are not forwarded upstream

Transition from a research project to a software editor

- Clean process ?
 - How to close a bug ?
 - How to remove a deprecated feature from the language ?
 - How to handle major and minor releases ?
 - How to integrate a new feature into the language ?
 - ...

Transition from a research project to a software editor

- Example: How to integrate a new feature ?
 - Write a SEP – Scilab Enhancement Proposal
 - What is it supposed to do ?
 - What would be the profile of the function ? (when applies)
 - How is it going to work ?
 - What is the expected behaviour with other existing functions ?
 - Which version is targeted ?
 - Validation

Transition from a research project to a software editor

- Example: How to integrate a new feature ? - 2
 - The implementation
 - The documentation
 - The unitary tests
 - The integration



Thanks for your attention



www.scilab.org

