Libérer votre application graphique Java avec Jython

Jean-Michel GLORIAN

Conférence COMPIL du 25/11/10
Langages de scripts

Plan

- Pourquoi utiliser un langage de script dans une application graphique ?
- Le langage de script Jython
- HIPE : Un exemple d'utilisation de Jython
- Conclusion et Perspective

Pourquoi utiliser un langage de script dans une application graphique ?

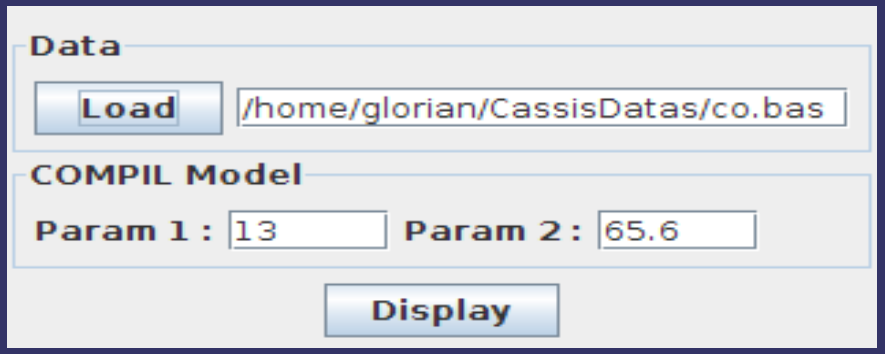
- Démonstration par l'exemple :
  - un chercheur
  - un informaticien
  - une application graphique

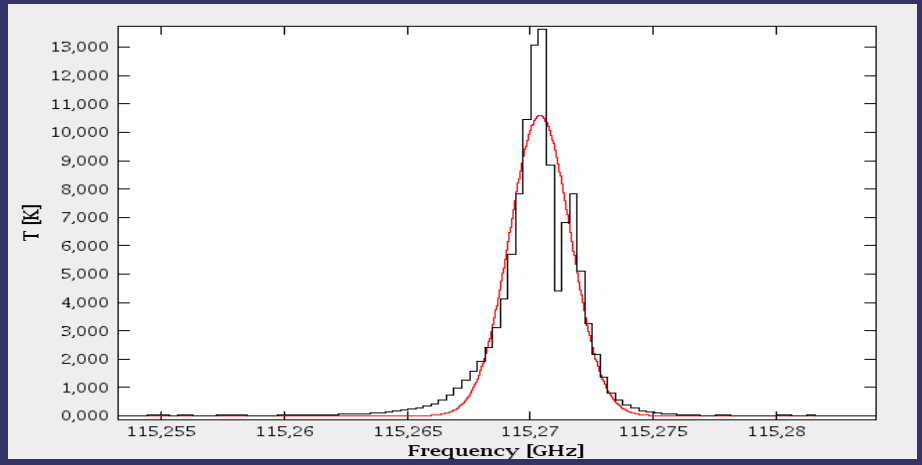
Utilisation de l'application graphique

Le chercheur

Il sélectionne un fichier de données
Il entre les 2 paramètres du "modèle COMPIL"

Tant que le modèle ne "colle pas bien" aux données
il change les paramètres
et ré-affiche les courbes (modèle + données)





Une application graphique

Fonctions développées par l'informaticien

- Affichage de saisie des 3 paramètres
  - 1 fichier de données
  - 2 paramètres pour le modèle
- Une méthode CalculeModel(param1, param2)
- Affichage des 2 courbes
  - les données
  - le modèle

L'informaticien

Une demande d'évolution

Ok, mais j'aimerais faire
varier automatiquement
ces 2 paramètres
et
j'aimerais qu'il m'affiche
le meilleur modèle par rapport
aux données

Nouveau code développé

- Pour chaque paramètre, 3 valeurs :
  Min, Max et Pas

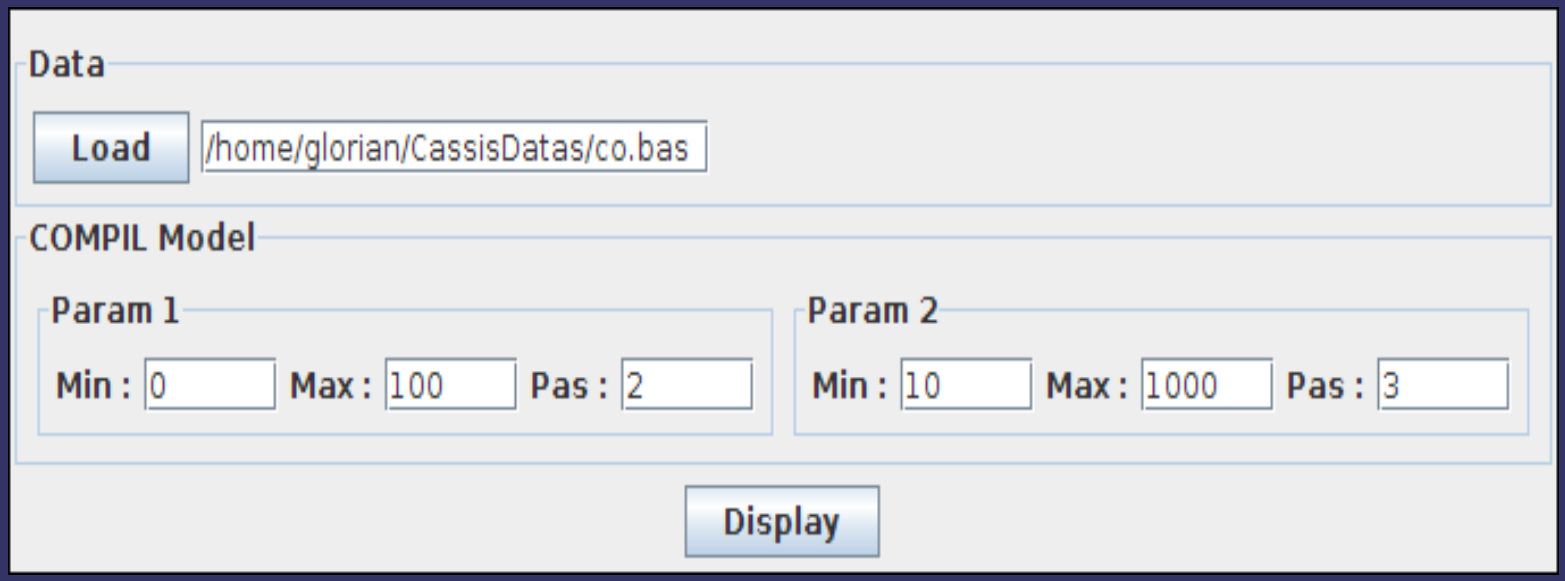
- Affichage de saisie des 7 paramètres
  - 1 fichier de données
  - 6 paramètres pour le modèle

- Pour val1 (param1Min, param1Max, param1Pas
  Pour val2 (param2Min, param2Max, pas)
  Modele_Courant = CalculeModel(val1, valt2)
  ajouter Modele_Courant à la liste des modèles

- modèle = SelectionMeilleurModele(données, modèles)

- Affichage des 2 courbes (les données et le modèle)

Modification de l'application graphique



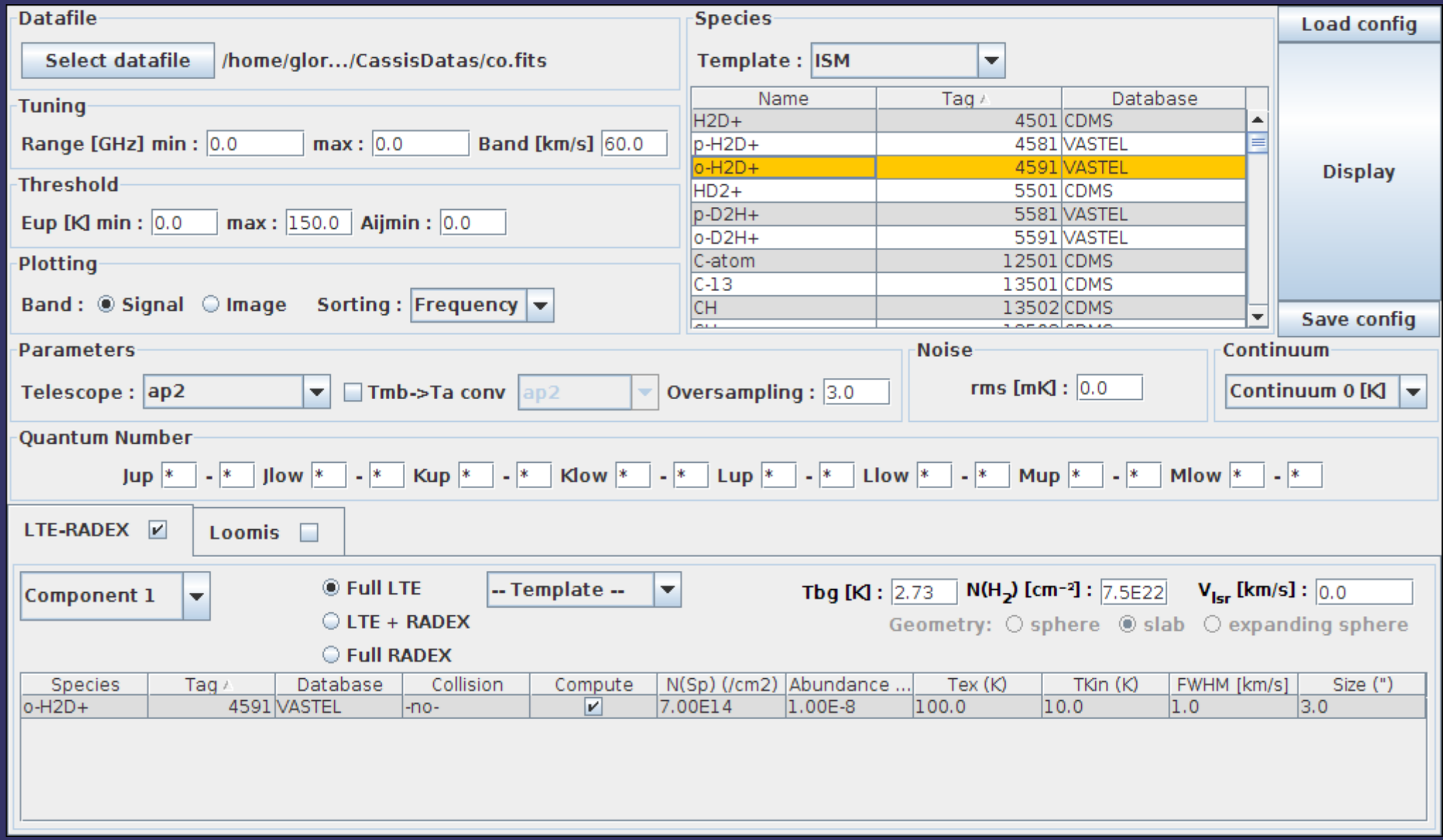
Le problème

- Sauf que le chercheur aimerait

  - faire varier ces paramètres de différentes façons (linéaire, sinusoïdale, …)
  - expérimenter plusieurs méthodes pour sélectionner le meilleur spectre
  - exclure des valeurs non désirables sur les paramètres suivant les données à comparer

et en plus ...

… dans un cas réel, le nombre de paramètres est plus grand et plus diversifié :



Une Solution

- Intégrer un langage de script dans l'application
- Le chercheur
  - écrit ses propres scripts
  - a accès à toutes les fonctionnalités du logiciel
  - écrit lui même ses boucles
  - peut contrôler très finement la valeur des paramètres du modèle
  - peut tester des algorithmes de recherche du meilleur modèle

Intégration d' un langage de script : quelques idées et réflexions

- L'utilisateur devient développeur
  - Il ajoute ses propres fonctionnalités en utilisant le code déjà existant
- Le développeur expose son code
  - incitation à fournir un code robuste et à avoir des noms de méthodes compréhensibles
- Le script permet d'automatiser des tâches répétitives
  - Cela oblige le développeur à bien séparer la vue et le modèle
- Le développeur peut générer des scripts lors des cliques de souris de l'utilisateur

Qu'est ce que Jython ?

- Java + Python
- Interpréteur Python écrit en Java.
  - Scripting : Exécution de code Python durant le fonctionnement d'un programme Java
  - Création de classes python qui étendent les classes Java
- Autres utilisations
  - Prototypage : test d'algorithme, ...
  - Investigation dans Java : print(java.util.Random), ...
  - Débogage : affichage des points d'une courbe, …
- Projet actif Jython 2.5.1 (26/09/09), release candidate V. 2.5.2 RC (24/10/10)

Comment intégrer Jython dans son application JAVA?

- Java SE 6.0 intègre la possibilité d'utiliser des moteurs de scripting
- Pour avoir Jython dans son application Java :
  - Ajouter jython.jar dans le classpath
  - Récupérer l'interpréteur Jython

```
ScriptEngineManager manager = new ScriptEngineManager();
ScriptEngine moteur = manager.getEngineByName("jython");
```

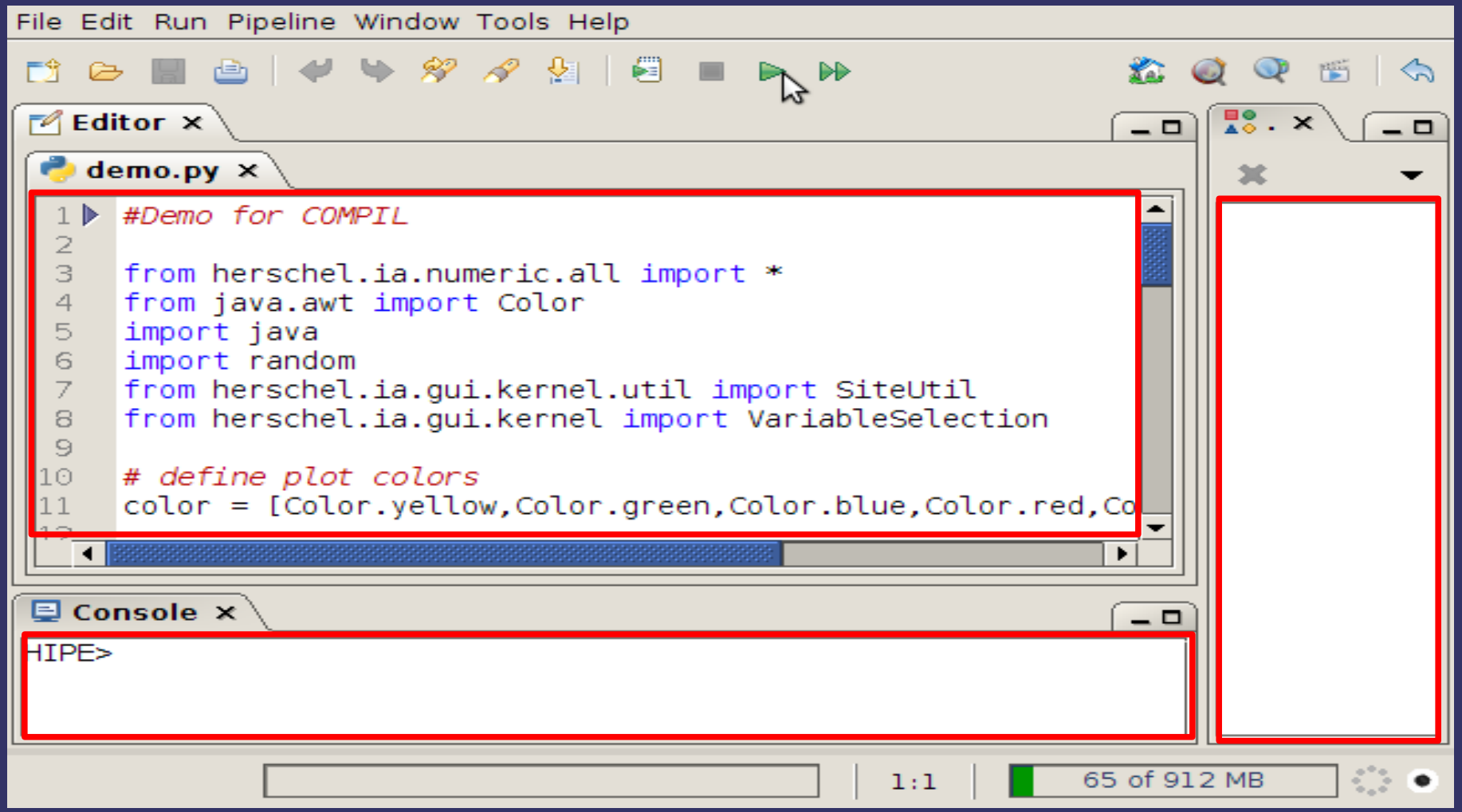
  - Évaluer ligne par ligne le script

```
moteur.eval( une_ligne_du_script );
```

Exemple d'utilisation de Jython : HIPE

- Herschel Interactive Processing Environment
  - Logiciel de traitement de données de l'observatoire spatial Herschel
  - Maître d'œuvre : l'Agence Spatiale Européenne
  - Développé en Java avec un noyau Eclipse et embarque un interpréteur Jython

Fenêtre principale de HIPE



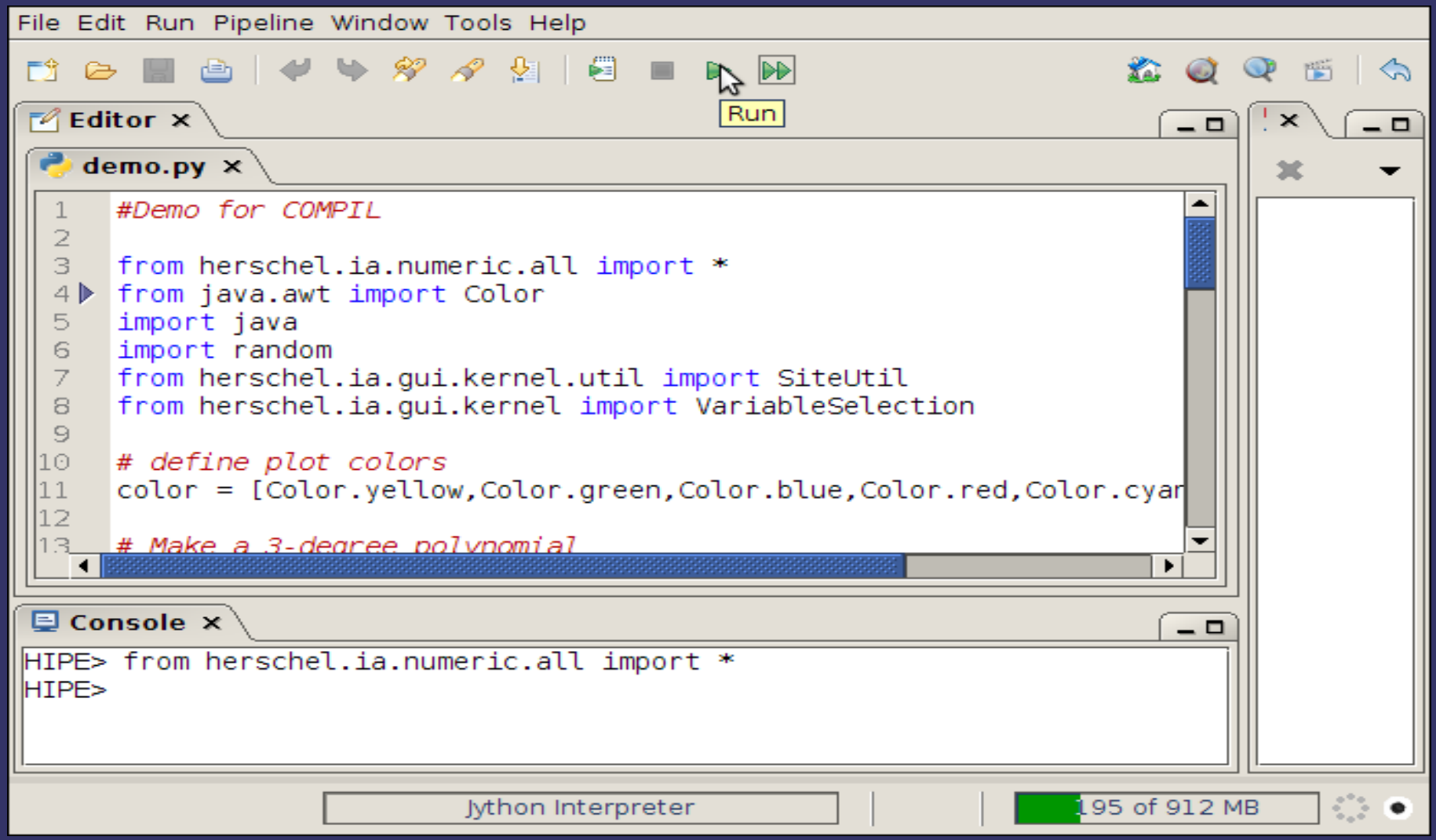
Exemple d'un script Jython dans HIPE (1/2)

```
#Demo for COMPIL

from herschel.ia.numeric.all import *
from java.awt import Color
import random
            ...
# définition des couleurs des courbes
color = [Color.yellow,Color.green,Color.blue,Color.red,Color.cyan]

# Création d'une fonction polynomiale de degré 3
x,y = Double1d(25), Double1d(25)
f = 1.2 + 0.5 * x + 0.13 * x**2 + 0.4 * x**3

for i in range(25):
    y[i] = f[i] + 12 * random.gauss(0, 1)  # 12 = noise
```

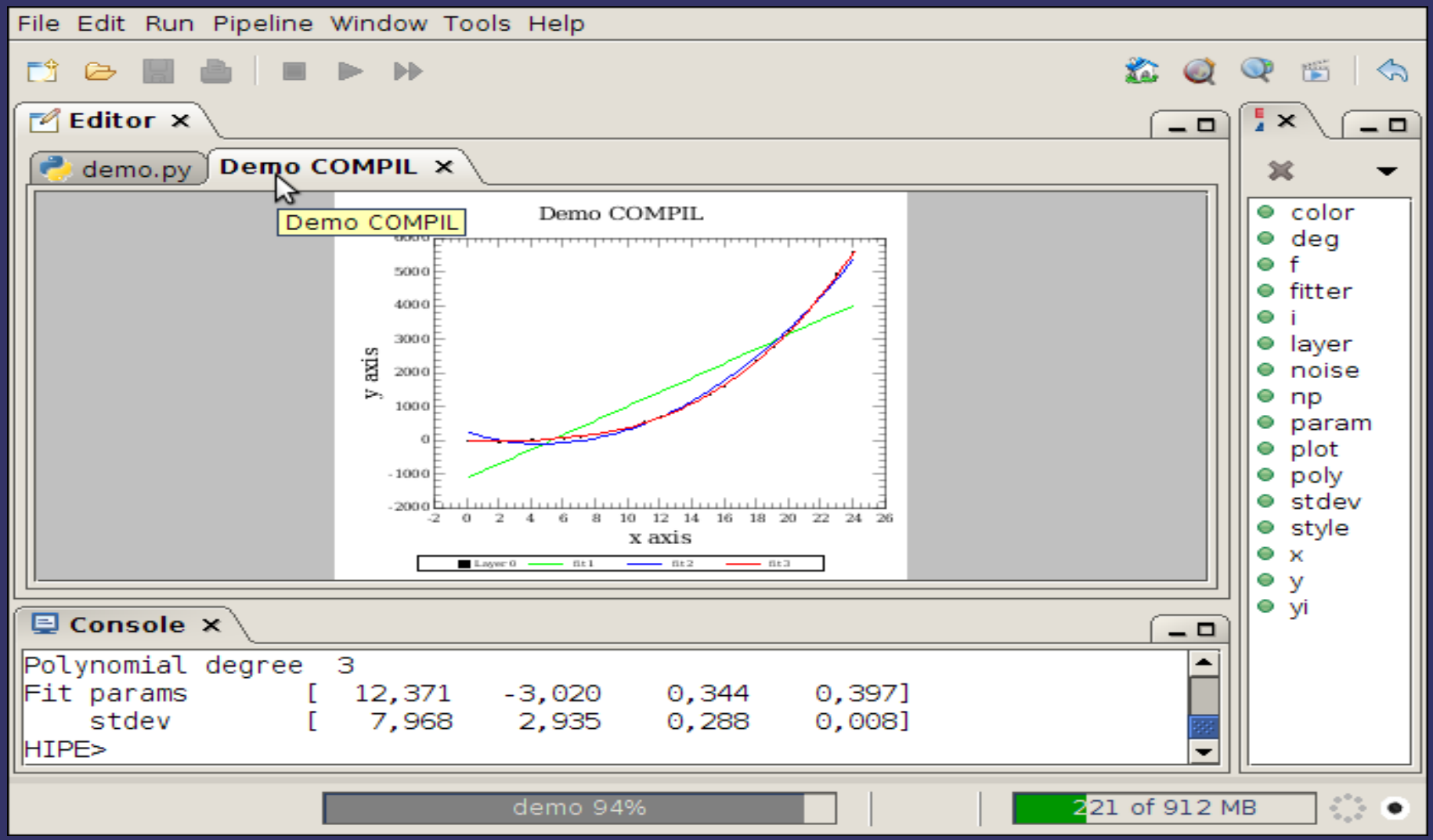
Exemple d'un script Jython dans HIPE (2/2)

```
style=Style( line=Style.NONE, symbol=Style.FSQUARE)
plot = PlotXY( titleText= "Demo COMPIL" )
plot.addLayer( LayerXY( x, y, style=style ) )

for deg in range(1,4) :
   poly = PolynomialModel( deg )
   fitter = Fitter( x, y, poly )
   layer = LayerXY( x, fitter.calc(x), name="fit " + str(deg), color=color[deg])
   plot.addLayer( layer )

   print "Polynomial degree ", deg
   print "Fit params      ", fitter.getParam()
   print "     stdev      ", fitter.getStandardDeviation()

SiteUtil.getEditorArea().open(VariableSelection('Demo COMPIL', plot)
```

Exécution de la 1ère instruction du script



Résultat du lancement du script



Conclusion et perspective

- Un langage de script dans votre application
  - rendra votre application plus puissante
  - influencera positivement la qualité de code de votre application
  - sera très utile dans les logiciels de traitement de données, dans les logiciels de test de matériel, …
  - libérera l'utilisateur de l'interface graphique
- Étant auto convaincu, l'auteur de ce slide intégrera Jython dans son application (CASSIS) ...

Liens

JYTHON : http://www.jython.org/

Mr DOUDOUX sur Developpez.com :
http://jmdoudoux.developpez.com/cours/developpons/java/chap-scripting.php

HIPE :
http://herschel.esac.esa.int/HIPE_download.shtml

JAVA :
http://www.oracle.com/technetwork/java/javase/overview/index.html