
FORMATION À SCILAB (NIVEAU AVANCÉ)

PROGRAMME DE FORMATION

Objectifs : Savoir utiliser SCILAB pour le calcul numérique, le traitement du signal, et la modélisation et la simulation des systèmes

Public concerné : Ingénieurs en traitement du signal, électronique ou informatique, ou toute personne ayant des notions de base en informatique et des besoins en calcul numérique

Prérequis : Notions de programmation et de traitement du signal (transformée de Fourier, etc.)

Durée : Trois jours

Matériel nécessaire pour suivre la formation : PC portable sous Windows, avec Scilab 5.5.0 (ou version plus récente) installé (pour pouvoir faire les exercices pratiques). Les participants peuvent aussi utiliser Linux ou Mac OS s'ils le souhaitent, mais ils devront alors vérifier au préalable que leur installation de Scilab est bien fonctionnelle (en particulier que les graphiques fonctionnent bien : par exemple la commande `plot(1:10)` doit ouvrir une fenêtre graphique).

Inscription et demande d'informations :

<http://www.tsdconseil.fr/formations/formulaire>

Informations pratiques, tarifs :

<http://www.tsdconseil.fr/formations/infos>

PARTIE 1 (6H) I - INTRODUCTION À SCILAB

Durant cette première partie, nous allons faire un tour d'horizon de l'environnement SCILAB, et des principales fonctions (calcul scalaire et matriciel, fonctions graphiques, programmation).

Introduction : Domaines d'application, description du contexte (par rapport aux autres applications de calcul)

Environnement : console SCILAB, aide en ligne, éditeur

Éléments de base du langage : Nombres réels et complexes, matrices et vecteurs, constantes, booléens, polynômes et fractions rationnelles, fonctions mathématiques usuelles

Programmation : scripts et fonctions, structures de contrôle, mise au point, entrées / sorties fichiers

Visualisation des données : représentations 2D et 3D, axes et légendes

PARTIE 2 (4H) II - ALGÈBRE LINÉAIRE

SCILAB est fourni avec des fonctions très avancées pour l'algèbre linéaire, permettant de traiter aussi bien des systèmes dits denses (à faible dimension), que des systèmes creux. De plus, le langage de SCILAB étant fondé sur les matrices, vous verrez que le calcul matriciel s'en trouve très simplifié...

Systèmes linéaires : systèmes inversibles, sur-déterminés, sous-déterminés, conditionnement, résolution au sens des moindres carrés, pseudo-inverse

Valeurs propres et vecteurs propres

Décompositions matricielles et applications (LU, QR, SVD, ...)

Problèmes à grande dimension

PARTIE 3 (2H) III - STATISTIQUES

Même si ce n'est pas sa spécialité, SCILAB inclut les fonctions les plus importantes pour simuler des processus aléatoires ou estimer des paramètres.

Fonctions de répartition

Mesures statistiques simples : moyenne, variance, médiane, ...

Simulation : générateurs congruents (limites), lois usuelles, permutations, chaînes de Markov

Estimation de paramètres : Estimation au maximum de vraisemblance / moindre carrés. Cas particulier : régression linéaire

Matrices et vecteurs

Matrices: Polynômes
zeros Sensibilité
ones Représentations 3D : surfaces
eye Maillages rectangulaires
rand

```
Exemple
--> A =
A =
[0. 0.
 0. 0.
 25.16]

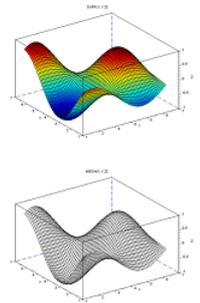
--> p =
p =
[1. 1.
 1. 1.
 1. 1.]

--> A :
A =
[1. 1.
 1. 1.
 1. 1.]

--> p =
p =
[1. 1.
 1. 1.
 1. 1.]

--> Z =
Z =
[1. 1.
 1. 1.
 1. 1.]

--> surf(X,Y,Z); // ou mesh(X,Y,Z)
```



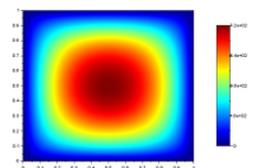
Systèmes carrés inversibles (r = n = m)

Deux méthodes: Valeurs propres, Résoudre

Opérateur: Matrices creuses : Application à l'équation de la chaleur
Méthode: Généralisation en 2D
Complexité

$$A = \begin{pmatrix} I & B & I & & \\ & \ddots & I & B & I \\ & & & \ddots & \\ & & & & I & B & I \\ & & & & & & \ddots & \\ & & & & & & & I & B & I \end{pmatrix}, \text{ avec } B = \begin{pmatrix} 1 & -4 & 1 & & \\ & \ddots & & \ddots & \\ & & & & 1 & -4 & 1 \\ & & & & & & \ddots & \\ & & & & & & & 1 & -4 & 1 \end{pmatrix}$$

- Temps de calcul en utilisant une matrice creuse et un maillage 100 x 100 du plan : 4 secondes
- Nombre d'opérations si on avait utilisé une matrice pleine : $(100 \times 100)^3 \sim 1000$ milliards!



Lois de probabilité

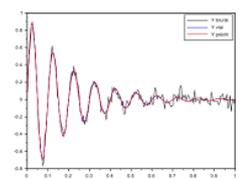
Loi normale: Mesures statistiques : covariance
Loi normale: Estimation de la matrice de covariance

Co-v.a: Estimation de paramètres
Re: Exemple avec datafit

Identification des paramètres (ω, ρ) du modèle : $y = \sin(\omega x)e^{-\rho x} + \epsilon$

```
function err = G(p,z)
x = z(1);
y = z(2);
omega=p(1), rho=p(2);
yp = sin(omega*x)*exp(-rho*x);
err = y - yp;
endfunction

// y contient les observations
--> x = (0:1/fe:1);
--> Z = [x ; y];
--> p = datafit(G, Z, [1;1]);
// p(1) = omega
// p(2) = rho
```



PARTIE 4 (5H) IV - TRAITEMENT DU SIGNAL

Nous allons voir comment SCILAB peut vous aider pour l'analyse des données expérimentales (analyse fréquentielle, corrélations, etc.) et la conception et l'analyse de filtres numériques (FIR, IIR).

Signaux numériques Échantillonnage, transformée de Fourier

Représentation des systèmes linéaires discrets : équation aux différences, fonction de transfert, représentation d'état (*state/space*). Analyse (temporelle, fréquentielle, pôles / zéros)

Conception des filtres numériques : FIR (fenêtrée, Remez), IIR (à partir de prototypes analogiques, factorisation en sections du second ordre), quantification

Interpolation : Interpolations linéaire, splines, interpolations 2D

Estimation spectrale : Méthodes non paramétriques (périodogrammes) / paramétrique (ARMA)

Autres fonctions : Filtre de Hilbert, filtre de Kalman

Signaux numériques

Transformée de Fourier discrète

X=fft Signaux numériques

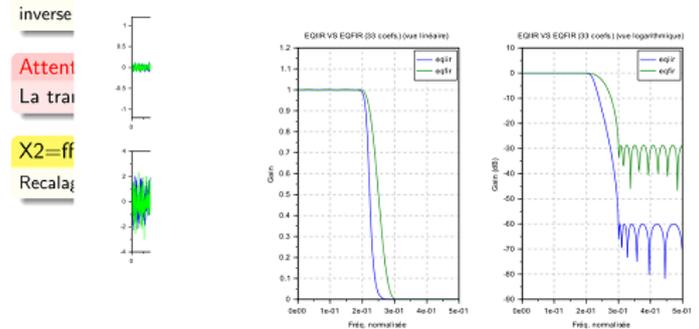
Transfc Produit de corrélation

Exemp Conception IIR : eqiir

• Id Comparaison eqiir VS eqfir

• C: Filtre IIR : 6 SOS + 1 section 1^{er} ordre ~ 33 MAC / éch.

Exemp Équivalent FIR : hfir=eqfir(33,[0 .2;.3 .5],[1 0],[1 1]);



PARTIE 5 (4H)

V - CALCUL DIFFÉRENTIEL, INTÉGRATION ET MODÉLISATION

La simulation des systèmes physiques nécessite souvent la résolution numérique d'équations différentielles : nous allons faire un tour d'horizon des fonctions correspondantes dans SCILAB. Ensuite, nous allons voir les fonctions principales de la bibliothèque d'optimisation numérique, qui permet de trouver des solutions optimales à divers problèmes pratiques (trajectoires optimales, affectation de ressources, etc.).

Calcul différentiel

Intégration numérique

Équations différentielles : ODE (premier ordre et ordre supérieur), détection de racines, aperçu sur les autres solveurs,

Optimisation : cas général, fonctions différentiables, problèmes à contraintes fixes / linéaires / non linéaires,

Recherche de racines

Équations différentielles ordinaires (ODE)

Fonction SCI AR n^{de}

y=ode(t Recherche de racines

Paramètres variables

Optimisation

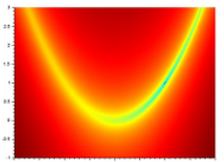
• S Problèmes sans contraintes : exemple

• C Fonction de Rosenbrock

```
function [y] = f1(x)
y = (1-x(1))^2 + 100*(x(2)-x(1)*x(1))^2;
endfunction
function [y,g,ind] = f2(x,ind)
y = f1(x);
g = numerivative(f1,x);
endfunction
function [y,g,ind] = f3(x,ind)
y = f1(x);
g(1)=-2+2*x(1)-400*...;
g(2)=200*(x(2)-x(1)**2);
endfunction
```

```
x1 = fminsearch(f1, x0);
x2 = optim(f2, x0);
x3 = optim(f3, x0);
```

$$f(x_1, x_2) = (1 - x_1)^2 + 100(x_2 - x_1^2)^2$$



Méthode	Temps	Précision
fminsearch	2.4 s	1,5e-5
optim / gradient numérique	78 ms	1,4e-8
optim / gradient fourni	< 1 ms	< εmach

PARTIE 6 (1H)

VI - AUTRES FONCTIONS (APERÇU)

XCOS : Simulateur et éditeur graphique de schéma bloc (systèmes discrets, continus, implicites)

Autres bibliothèques

Références, remarques finales et questions

