

TRAVAUX PRATIQUES
RESEAX DE NEURONES TYPE « Feedforward »
Simulation en utilisant Scilab-ANN

CONTEXT

L'objectif de ce TP est de simuler un réseau de type feedforward (voir cours <http://www.esiee.fr/~alanit>) et d'entraîner ce réseau pour implémenter une fonction spécifique. La simulation sera effectuée sous Scilab <http://www-rocq.inria.fr/scilab/> avec la boîte à outils de réseau neurale ANN http://www.scilab.org/contrib/index_contrib.php?page=download.php&category=MODELING%20AND%20CONTROL%20TOOLS

I. FAMILIARISATION AVEC SCILAB

Exécuter Scilab dans un terminal : `>>scilab`

Découvrez les différentes fonctionnalités de Scilab (voir demo dans l'onglet « help »).

II. INSTALLATION DU TOOLBOX ANN

Charger le fichier .zip sur la page IA.

Décompresser le fichier TP_IA0708.zip dans votre répertoire de travail, TPIA par exemple. S'il n'est pas déjà installé, installer localement le toolbox « ANN_Toolbox_0.4.2 » : dans la fenêtre de Scilab tapez la commande suivante pour charger les fonctions *.sci qui se trouve dans le chemin de ANN_Toolbox_0.4.2 «path»

```
exec(path+'ANN_Toolbox_0.4.2\loader.sce');
```

Lire dans le « help /Artificial neural networks» les fichiers « ANN », « ANN_FF » et ANN_GEN.

III. IMPLEMENTATION D'UNE FONCTION NON-LINEAIRE (2D) PAR UN RESEAU DE NEURONES

La fonction à implémenter est la suivante :

$y = \sin^2(\pi x_1) * \sin^2(\pi x_2)$ avec $\mathbf{x}_1 = \mathbf{x}_2 = [1 : 0.01 : 1]$

III.1. GENERATION DES DONNEES D'APPRENTISSAGE ET DE TEST

Implémenter les fonctions suivantes en code Scilab :

- « x.sci » (fonction $[\mathbf{x}, \mathbf{xs}] = \mathbf{x}_i()$) pour la génération d'un vecteur d'entrée \mathbf{x} (avec $\mathbf{x}_1 = \mathbf{x}_2$).
- « y.sci » (fonction $[\mathbf{t}, \mathbf{xs}] = \mathbf{y}(\mathbf{x})$) pour la génération des vecteurs cibles t_1, t_2, \dots, t_n^2 , où $n = \text{size}(\mathbf{x}, 2)$. La matrice \mathbf{xs} est générée pour l'utiliser dans la fonction de l'apprentissage (Backpropagation ANN with momentum « ann_FF_Mom_online.sci ») :

```
xs=zeros(2,n*n);  
// number of samples for the two variables=n*n  
// size(xs(1,:,:),:): (1. n*n)  
// size(xs(2,:,:),:): (1. n*n)  
  
for i=1:n,  
    for j=1:n,  
        xs(1,(i-1)*n+j)=x(i);  
        xs(2,(i-1)*n+j)=x(j);  
    end  
end
```

III.2 INITIALISATION ET APPRENTISSAGE DU RESEAU

Les fonctions utilisées pour l'apprentissage du réseau sont appelées par un fichier exécutable « train.sce ». Elle crée le réseau final. Le nombre d'itérations pour l'apprentissage est contrôlé par un paramètre « epoques ». Ce paramètre définit combien de fois la boucle d'apprentissage est exécutée, par exemple, epoques = 50. Après que les époques dans une boucle sont finies, le réseau résultant est sauvegardé seulement si une amélioration est obtenue. lp définit le facteur par lequel le taux d'apprentissage est adapté dans chaque boucle.

- « init.sci » (fonction [W,N,Delta_W_old,lp]=init(x,t)) pour initialiser le réseau. La matrice de poids du réseau W peut être initialisée en utilisant la fonction « ann_FF_init.sci » et ensuite « ann_FF_Mom_online.sci » de toolbox ANN :

```
// Topologie du réseau
N = [2,8,6,1];
// Entrées : x
// Sorties désirées (targets) : t
// Paramètres d'apprentissage
lp = [0.8, 0.01, 0.8, 0.1];
// initialisation des poids aléatoires entre:
r = [-10,15];
rb = r;
// Initialisation de la matrice des poids
W = ann_FF_init(N,r,rb);
Delta_W_old = hypermat(size(W));
epochs=1;
// Exécuter le réseau pour un epoch
[W,Delta_W_old] = ann_FF_Mom_online(x,t,N,W,lp,epochs,Delta_W_old);
```

- «train.sci » (fonction [W,N,Delta_W_old,lp]=train(xs,epochs,W,N,Delta_W_old,lp)) pour entraîner le réseau. Le parameter epochs peut être initialisé dans le programme principal « main_train.sce » (voir ci-dessous). La matrice de poids du réseau W peut être calculée en utilisant « ann_FF_Mom_online.sci » de toolbox ANN.

Le réseau est alors sauvegardé dans le fichier "network.dat", y compris sa configuration, la matrice de poids, les paramètres d'apprentissage.

```
//save updated network parameters
save (path+'network.dat', W, N, Delta_W_old, lp);
```

- Tester le réseau avec les mêmes données d'apprentissage avec « y = ann_FF_run(x,N,W) »
- Créer un programme Scilab exécutable « main_train.sce » appelant toutes les fonctions précédentes:

```
//Program main_train
set old_style on;// initialisation de graphique
lines(0); //pour un défilement sans arrêt pour le défilement continu de texte
path=get_absolute_file_path('main_train.sce');
getf(path+'xi.sci');// charger la fonction 'x.sci' qui se trouve dans « path »
getf(path+'y.sci');// charger la fonction 'y.sci' qui se trouve dans « path »
getf(path+'init.sci');
getf(path+'train.sci');
exec(path+'ANN_Toolbox_0.4.2\loader.sce');// charger les fonctions *.sci qui se trouve dans «path»
[x,xs]=xi();
n=size(x,2);
[t]=y(x);
epochs=50;
printf("Training, be patient please!\n");
[W,N,Delta_W_old,lp]=init();
[W,N,Delta_W_old,lp]=train(xs,epochs,W,N,Delta_W_old,lp);
y = ann_FF_run(x,N,W);
// sauvegarde des entrées, cibles et paramètres du réseau
path=get_absolute_file_path('main_train.sce');
save (path+'x.dat',x);// Sauvegarder x dans votre chemin sous le nom « x.dat »
save (path+'t.dat',t);// Sauvegarder t dans votre chemin sous le nom « t.dat »
save (path+'network.dat', W, N, Delta_W_old, lp);
// sauvegarde des sorties du réseau
save (path+'result.dat', y);
```

- Sauvegarder puis exécuter « main.sce » dans votre répertoire de travail.