HW Assignment 1 (Due by 10:30am on Sep 18)

1 Theory (80 points)

1. **[Linear Regression, 20 points]**
   Prove that the least square solution to polynomial curve fitting satisfies the set of $M+1$ linear equations shown on slide 17 in lecture 1.

2. **[Perceptron, 20 points]**
   Consider a training set that contains the 4 examples below i.e., the truth table of the logical XOR function. Prove that the perceptron algorithm does not converge on this dataset. Do not forget the bias feature $x_0 = 1$.

   \[
   \begin{array}{|c|c|c|}
   \hline
   x_1 & x_2 & t \\
   \hline
   0 & 0 & 0 \\
   0 & 1 & 1 \\
   1 & 0 & 1 \\
   1 & 1 & 0 \\
   \hline
   \end{array}
   \]

3. **[Logistic Regression, 20 points]**
   Prove that the gradient (with respect to $w$) of the negative log-likelihood error function for logistic regression corresponds to the formula shown on slide 48 from lecture 1:

   \[
   \nabla_w E(w) = \sum_{n=1}^{N} (h_n - t_n)x_n^T
   \] (1)

4. **[Softmax Regression, 20 points]**
   Prove that the gradient (with respect to $w_k$) of the negative log-likelihood error function for regularized softmax regression corresponds to the formula shown on slide 57 from lecture 1, for any class $k \in [1..K]$:

   \[
   \nabla_{w_k} E(w) = -\sum_{n=1}^{N} (\delta_k(t_n) - p(C_k|x_n))x_n^T + \alpha w_k^T
   \] (2)

2 Implementation (100 points + 20 bonus)

Implement the softmax regression model, as explained on the course web site. Make sure that you organize your code in folders as shown in the table below. Write code only in the 3 files indicated in bold. The points will be allocated as follows:

- 10 points for implementing the numerical gradient function.
- 60 points for a correct implementation of the cost and the gradient that lead to the expected accuracy.
• Make sure you use the average of the negative log-likelihood in the cost function (i.e. divide by the number of training examples).

• 30 points for full vectorization.

• 20 bonus points for rewriting in vectorized Matlab the game of life code shown in the function `iterate_2` in the NumPy tutorial below: [http://www.labri.fr/perso/nrougier/teaching/numpy/numpy.html](http://www.labri.fr/perso/nrougier/teaching/numpy/numpy.html)

3 Submission

Turn in a hard copy of your homework report at the beginning of class on the due date. Electronically submit a directory that contains only the 3 required files. Make sure your code runs correctly when used in the architecture shown above. Create a gzipped, tar ball archive of your directory, and upload it on Blackboard by the due date.

For example, if the folder containing the required files is ‘hw01’, creating the archive can be done using the following commands:

```
> tar cvf hw01.tar hw01
> gzip hw01.tar
```

These two steps will create the file ‘hw01.tar.gz’ that you can upload on Blackboard. It is also acceptable to submit a zip file ‘hw01.zip’ created with the zip command.

Please observe the following when handing in homework:

1. Structure, indent, and format your code well.

2. Use adequate comments, both block and in-line to document your code.

3. On the theory assignment, clear and complete explanations and proofs of your results are as important as getting the right answer.

4. Make sure your submitted files work as expected in the folder structure shown above, e.g. calling `softmaxTrain` followed by `softmaxPredict` should lead to the correct results, with no error messages.