Aplicaţie:

Crearea şi antrenarea unei reţele neurale pentru aproximarea funcţiei

$$f: [-1,1] \to R, f(x) = \sin(2\pi x).$$

Se dă mulţimea de antrenare:

```
X = -1:0.05:1;
t = sin(2*pi*X)+0.1*randn(size(X));
```

Cerinţe:

1. Creaţi o reţea având un singur nivel ascuns cu 20 de perceptroni pentru aproximarea funcţiei f.
2. Antrenaţi reţeaua, folosind algoritmul backprop cu momentum şi rată variabilă, cât timp valoarea funcţiei de performanţă/eroare (MSE) este mai mare decât 0.001 şi numărul de epoci este mai mic de 500.
3. Să se reprezinte grafic eroarea (MSE) după fiecare epocă.
4. (**Exerciţiu suplimentar**) Să se reprezinte grafic eroarea (MSE) la fiecare moment de timp (se poate folosi variabila *cputime*).
5. (**Îmbunătăţirea generalizării**) Implementaţi algoritmul de antrenare "Bayesian Regularization backpropagation" folosind funcţia din Matlab 'trainbr' (net.trainFcn = 'trainbr'). Precizaţi numărul efectiv de parametri folosiţi de reţea. Precizaţi câţi perceptroni sunt (de fapt) necesari pe nivelul ascuns pentru aproximarea funcţiei f.
6. (**Îmbunătăţirea generalizării**) Folosiţi în procesul de antrenare tehnica 'Early stopping' şi funcţiile de împărţire a mulţimii de exemple/de învăţare în cele trei submulţimi (mulţimea de antrenare, mulţimea de validare şi mulţimea de testare): 'divideind', 'dividerand', 'divideint' şi 'divideblock'.
   Indicaţie: Spre exemplu, pentru a folosi tehnica 'Early stopping' şi funcţia de împărţire a mulţimii de exemple 'divideind', se setează următoarele câmpuri ale obiectului de tip reţea **net**:

   ```
   net.divideFcn = 'divideind';
   indici_antrenare = [1:4:m 3:4:m];
   indici_validare =  2:4:m;
   indici_testare = 4:4:m;
   net.divideParam.trainInd = indici_antrenare;
   net.divideParam.valInd = indici_validare;
   net.divideParam.testInd = indici_testare;
   ```

7. Dacă eroarea de validare creşte timp de un număr de iteraţii specificat (net.trainParam.max_fail) antrenarea se opreşte şi se returnează ponderile şi bias-urile corespunzătoare erorii de validare minime. Modificaţi numărul de iteraţii după care antrenarea este oprită.
8. Folosind punctul 3., să se compare performanţele algoritmilor de antrenare prezentaţi în fişierul Laborator 10. Care este cel mai adecvat rezolvării acestei probleme?
9. (**Exerciţiu suplimentar**) Studiaţi convergenţa algoritmilor de antrenare folosind programele demonstrative indicate în fişierul Laborator 10.